Context-Free Event Domains are recognizable

Eric Badouel, Philippe Darondeau, Jean-Claude Raoult

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Context-Free Event Domains are recognizable

Eric Badouel Philippe Darondeau and Jean-Claude Raoult

N° 2588

juin 1995

PROGRAMME 2

ISSN 0249-6399

Context-Free Event Domains are recognizable

Eric Badouel * Philippe Darondeau* and Jean-Claude Raoult*

Programme 2 — Calcul symbolique, programmation et génie logiciel
Projet Micas

Rapport de recherche n ° 2588 — juin 1995 — 56 pages

Abstract: The possibly non distributive event domains which arise from Winskel's event structures with binary conflict are known to coincide with the domains of configurations of Stark's trace automata. We prove that whenever the transitive reduction of the order on finite elements in an event domain is a context-free graph in the sense of Müller and Schupp, that event domain may also be generated from a finite trace automaton, where both the set of states and the concurrent alphabet are finite. We show that the set of graph grammars which generate event domains is a recursive set. We obtain altogether an effective procedure which decides from an unlabelled graph grammar whether it generates an event domain and which constructs in that case a finite trace automaton recognizing that event domain. The advantage of trace automata over unlabelled graph grammars is to provide for a more concrete and therefore more tractable representation of event domains, well suited to an automated verification of their properties.

Key-words: Event Domains, Trace Automata, Context-Free Graphs, Recognizability

(Résumé : tsvp)

This work was partly supported by the French P.R.C. *Modèles et Preuves*, by the H.C.M. Network *Express*, and by the Esprit B.R.A. *Asmics*.

*Email:{Eric.Badouel, Philippe.Darondeau, Jean-Claude.Raoult}@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Les domaines d'événements « context-free » sont reconnaissables

Résumé : Nous considérons la classe des « domaines d'événements ». Il s'agit des domaines de configurations de structures d'événements à conflit binaire ou de façon équivalente des domaines de configurations d'automates trace. Nous prouvons que tout domaine d'événements dont la réduction transitive de l'ordre sur les éléments finis est un graphe « context-free » au sens de Müller et Schupp peut être engendré par un automate trace fini. Nous montrons par ailleurs que l'ensemble des grammaires de graphes qui engendrent des domaines d'événements est récursif. Globalement nous disposons d'une procédure effective décidant si une grammaire de graphes non étiquetés engendre un domaine d'événements et qui construit dans l'affirmative un automate trace fini reconnaissant ce domaine. L'avantage des automates trace vis à vis des grammaires de graphes est de fournir une représentation plus concrète des domaines d'événements et donc *a priori* plus facilement utilisable en vue d'une vérification automatisée de leurs propriétés.

Mots-clé : Domaines d'événements, automates trace , graphes context-free, reconnaissabilité

Contents

1 Introduction **3**

2 Trace Automata and Conflict Event Domains **5**

3 Recognizable Event Domains **9**

4 Context-Free Event Domains **17**
4.1 Graph Grammars Generating Event Domains 17
4.2 Extracting a Trace Automaton from a Graph Grammar 25

5 Conclusion **33**

6 Appendices **34**
6.1 A Characterization of Context-free Event Domains 34
6.2 Deciding whether a Graph Grammar Generates a Conflict Event Domain 36
6.2.1 Transitive Reductions of Conflict Event Domains 37
6.2.2 Monadic Second Order Logic and Conflict Event Domains 39
6.2.3 An alternative decision for Axioms (C), (R), and (V) . . 44

1 Introduction

This study arises from the junction of two related trends in concurrency theory, namely the comparison of semantic models and the concern for expressivity. The first trend, best represented in [NW93], aims at charting embeddings and equivalences between categories of models of concurrency, thus comparing their intrisic power of representation. For instance, we showed in [BD93] that separated trace automata are equivalent with saturated trace nets and correspond at the level of domains of configurations with event structures with binary conflict. The resulting circuit between models preserves behaviours i.e. domains of configurations, and it also preserves finiteness. Such behaviour preserving correspondences between models help saving the effort of constructing explicitly several interpretations for a given programming language while remaining free to choose the most convenient model for analysing such and such

semantic feature (e.g. nets for causality, automata for cycles and termination, etc.). The second trend, appeared in [deS84] and [BBK87a] and amplified in [Vaa92], aims at measuring the extension of one or several (fragments of) programming languages interpreted in a given model. Possible scales of measure are the polynomial and arithmetic hierarchies of sets or the Chomsky hierarchy of languages, applied to the objects of the models viewed as sets or languages. For example, it was shown in [deS84] that a combined use of unguarded recursion and of operational rules in de Simones's format yields enough power for assigning a finite expression to any recursively enumerable transition system considered up to strong bisimulation. We show in this paper that any event domain which is generated from a context-free graph grammar can also be generated from some finite trace automaton. In order to establish that result, we construct an effective (partial) mapping of unlabelled graph grammars into finite trace automata, preserving domains of configurations.

An event domain is the set of configurations of an event structure with binary conflict, ordered by set inclusion, and it is in particular a coherent and finitary Scott domain [Win80]. As such, an event domain is totally determined by the transitive reduction (i.e. Hasse diagram) of the order on its finite elements. An event domain is said to be context-free when that diagram may be generated from a context-free graph grammar or equivalently when finitely many types of non isomorphic connected components are left after removing all nodes at depth less that some arbitrary constant [MS85]. We will show that the sub-family of unlabelled graph grammars generating event domains forms a recursive set. The mapping of that recursive set into the set of finite trace automata induces regular labellings on Hasse diagrams of context free event domains, where vertices are mapped to a finite set of states and arcs are mapped to a finite concurrent alphabet. The outcome is a more concrete and tractable presentation of event domains, well suited to an automated verification of their properties.

Let us add a few words about the background of this work. Context-free graphs, which may in general present cycles, have been studied at depth. For instance, they are known to coincide with the algebraic graphs of [Cou90] and with the pattern graphs of [Cau92]. On the other hand, recursively defined and effective domains have been thoroughly investigated [Smy77], but no attention has been paid so far to families of domains with a regular structure, such

as the context-free event domains. The present work is a limited attempt in this direction. Context-free event domains are much more general than the so-called context-free processes which are usually considered in the litterature on bisimulation [BBK87b]; in our opinion, the associated event structures provide a tractable alternative to the recursively defined event structures of [GL91].

The rest of the paper is organized as follows. Section 2 recalls the basic relationships between conflict event domains, event structures with binary conflict, and trace automata. Section 3 introduces a general definition of recognizable (but not necessarily context-free) event domains, and reports our early work towards their order theoretic characterization. We focus on context-free event domains in Section 4, which contains the main results. We give in an appendix two alternative procedures deciding whether a graph grammar defines a conflict event domain. The first procedure relies on Müller and Schupp's theorem, which states the decidability of the monadic theory of context free graphs [MS85]. The second procedure is more explicit and relies solely on the computation of fixed points of monotone operators on finite lattices.

2 Trace Automata and Conflict Event Domains

This section recalls the definitions of conflict event domains, event structures with binary conflict and trace automata, and the basic theorems stating their tight relationships. All the material of the section is borrowed from the litterature, especially from [Win80], [Win88], [Sta89a] and [Sta89b]. A *Scott domain* is an ω-algebraic and consistently complete partial order, and it is *finitary* if every finite element dominates finitely many elements.

> In an ordered set $(D, \leq)$, a subset X is *compatible* (notation: $X \uparrow$) if it has an upper bound, and two elements x and y are *compatible* (notation: $x \uparrow y$) if $\{x, y\}$ is compatible. A PO is *consistently complete* if every consistent subset has a least upper bound (X is *consistent*, or finitely compatible, if every finite subset of X is compatible). Consistently complete PO's coincide with bounded complete CPO's (a PO is a *CPO* if every directed subset X has a least upper bound and a CPO is *bounded complete* if every compatible pair has a least upper bound). A PO is

> *algebraic* if, for every element x, the set of finite elements smaller than x is directed and x is its least upper bound (y is finite if, for every directed X with least upper bound $\bigsqcup X$, $y \leq \bigsqcup X \Rightarrow \exists x \in X \,.\, y \leq x$). A PO is *$\omega$-algebraic* if it is algebraic and has countably many finite elements.

All the elements of a Scott domain are least upper bounds of sets of finite elements. In the particular case of an event domain, a finite element may be further decomposed into a set of indivisible grains of information, called events, each of which is obtained by passing through a specific class of projective prime intervals on some path leading to that element.

> We reserve the notation $[x, y]$ to the *prime* intervals, i.e. to the intervals such that $x \prec\!\!\!\cdot\, y$ (x is *covered* by y), meaning that $x < y$ and there is no element between them ($x \leq z \leq y \Rightarrow z = x$ or $z = y$). The *projective order* $\sqsubseteq$ on prime intervals is the least order relation such that $[x, y] \sqsubseteq [z, t]$ if $x = y \wedge z$ and $t = y \vee z$. The relation of *projectivity*, noted $\succ\!\!\!\cdot\!\prec$, is the equivalence on prime intervals generated by $\sqsubseteq$. An equivalence class of prime intervals is called an *event*. Intuitively, projective prime intervals represent the same increment of information.

The finite elements of an event domain are mapped bijectively to the associated sets of events, called *configurations*. This mapping may be extended by continuity to arbitrary elements, resulting in an isomorphism between every event domain and the associated set of configurations ordered by inclusion (this isomorphism maps the least upper bound of a compatible set of elements to the union of their associated configurations).

> Let us analyze the axiomatic requirements for the above representation in a finitary Scott domain, where we associate to every pair of finite elements x and y such that $x \prec\!\!\!\cdot\, y$ a transition $x \stackrel{a}{\rightarrow} y$ labelled by the projective class a of the prime interval $[x, y]$. In order that a finite element x may be represented by the set of event E_x occurring on an arbitrary sequence of transitions from the infimum to x, the following should hold:
> **Axiom JD:** *all the paths from the infimum to x have the same length and they are labelled by the same set of events, each of which occurs exactly once on each path.*
> In order that the function which maps y to E_y should be injective, the

transition system must be deterministic:
Axiom R: $[x,y] \succ\!\!\prec [x,y'] \Rightarrow \ y = y'$
In order that $E_{x \vee y} = E_x \cup E_y$, the following is finally required :
Axiom C: $[\ x \prec y \ \ \& \ \ x \prec z\ ] \ \& \ [\ y \uparrow z \ \ \& \ \ y \neq z\ ] \Rightarrow \ [\ y \prec (y \vee z) \ \ \& \ \ z \prec (y \vee z)\ ]$
Now, the configuration E_x associated with an arbitrary element x may be defined as the union of configurations E_y for finite approximations y of x .

A characteristic feature of *conflict* event domains, not taken into account in the above analysis of general event domains, is the following: in a conflict event domain, the relation of compatibility between elements or configurations is totally determined by a binary relation on events, called the conflict relation. Namely x and y are compatible ($x \uparrow y$) if and only if E_x and E_y do not contain conflicting events. By adding a specific axiom (V) encoding this feature, and dropping axiom (JD) which then becomes redundant, one ends up with Winskel's definition of conflict event domains.

Definition 1 (Conflict Event Domains) *A conflict event domain is a complete ω-algebraic partial order whose finite elements satisfy the following four axioms:*

Axiom F: $\{y \mid y \leq x\}$ is finite
Axiom C: $x \prec y$, $x \prec z$, $y \neq z$, $y \uparrow z \Rightarrow y \vee z$ exists & $y \prec (y \vee z)$, $z \prec (y \vee z)$
Axiom R: $[x,y] \succ\!\!\prec [x,y'] \ \Rightarrow \ y = y'$
Axiom V: $[x,x'] \succ\!\!\prec [y,y']$ & $[x,x''] \succ\!\!\prec [y,y'']$ & $x' \uparrow x'' \ \Rightarrow \ y' \uparrow y''$

In fact, one can prove (see for instance [Cur86]) that any two paths between two finite elements are equivalent by permutations, where the equivalence by permutations is the congruence (w.r.t. left and right concatenations) generated by the pairs $x \xrightarrow{a} y \xrightarrow{b} z \ \sim \ x \xrightarrow{b} y' \xrightarrow{a} z$. Moreover, one can also prove (see again [Cur86]) that axioms (F) and (C) imply that the order is consistently complete, thus a conflict event domain is a finitary Scott domain. Conflict event domains may be given a purely set theoretic representation in the form of (conflict) event structures defined as follows.

Definition 2 (Event Structures and their Configurations)
An event structure *is a triple* $(E, \#, \vdash)$ *where*
1. E *is a* countable *set of* events*;*

2. *$\#$ is a binary, symmetric and irreflexive relation on E, called the* conflict *relation;*
3. *let Con be the family of finite and conflict-free subsets of E then $\vdash$ is a subset of $Con \times E$, called the* enabling *relation, such that*
$$(X \vdash e \quad and \quad X \subset Y \in Con) \;\Rightarrow\; Y \vdash e$$

A configuration *is a subset $X \subset E$ which is*

1. conflict free*: $e\#e' \Rightarrow$ ($e \notin X$ or $e' \notin X$) and,*
2. secured*: $\forall e \in X \;\exists e_0 \ldots e_n \in X$ such that $e_n = e$ and $\{e_0, \ldots e_{i-1}\} \vdash e_i$ for $i = 1, \ldots, n$.*

Theorem 3 (G. Winskel)
i) The set of configurations $D(E)$ of an event structure E, ordered by inclusion, is a conflict event domain.
ii) Conversely, let D be a conflict event domain, then $(E_D, \#_D, \vdash_D)$ is an event structure with the following definitions.

1. *E_D is the set of events (i.e. classes of projective prime intervals) of the event domain D,*
2. *$a\#_D b$ if and only if there exist x, y, and z in D such that $[x,y] \in a$, $[x,z] \in b$, and $y \not\gamma z$,*
3. *let E_- be the representation function mapping elements of D to configurations, i.e. $E_x = \{[y,y']_{\prec} \,/\, y' \leq x\}$, then $X \vdash_D a$ for a finite and conflict free set of events X iff there exist some x and y in D such that $E_x \subset X$ and $[x,y] \in a$.*

Moreover, E_- is an isomorphism from D onto $D(E_D)$.

The above theorem yields an explicit connection between conflict event structures and conflict event domains. Let us now explain the relationship between conflict event domains and trace automata. Given a conflict event domain D, a labelled and acyclic transition system T_D may be derived from D by setting $x \xrightarrow{a} y$ in T_D when $[x,y] \in a$. Two events a and b are *independent*, written $a \|_D b$ when there exist x , y and z in D such that $[x,y] \in a$, $[x,z] \in b$, and $y \uparrow z$. It should be clear from the following definition that T_D becomes in this way a trace automaton over the concurrent alphabet $(E_D, \|_D)$.

Definition 4 (Trace Automata) *An automaton $\mathcal{A} = (A, Q, q_0, T)$ consists of a* countable *set A of actions, a set Q of states with initial state $q_0 \in Q$, and a transition relation $T \subset Q \times A \times Q$. A* trace automaton *$\mathcal{A} = (A, \|, Q, q_0, T)$ is an automaton whose alphabet comes equipped with a symmetric and irreflexive relation $\| \subset A \times A$, called the* independence relation*, such that the following conditions on transitions $p \xrightarrow{a} q \in T$ are satisfied:*

determinism : $(p \xrightarrow{a} q \ \wedge \ p \xrightarrow{a} r) \Rightarrow q = r$

commutativity : $(a \| b \ \wedge \ q \xrightarrow{a} r \ \wedge \ q \xrightarrow{b} s) \Rightarrow (\exists p \quad r \xrightarrow{b} p \ \wedge \ s \xrightarrow{a} p)$

Indeed, axiom R says exactly that T_D is deterministic, while axioms C and V entail commutativity.

Notation 5 *By an abuse of notation, we let $T_D = (T_D, \|_D)$ denote the trace automaton derived from the conflict event domain D.*

Observe that a trace automaton may be infinite (like T_D if D is infinite) and that it may present cycles (unlike T_D). Two trace automata which are finite and cyclic, resp. infinite and acyclic, may nevertheless generate two isomorphic domains of configurations as defined below.

> Let us fix notations and terminology. In a deterministic transition system $T \subset Q \times A \times Q$, $x \xrightarrow{a}$ is an abbreviation for $\exists y \,.\, x \xrightarrow{a} y$, and $x.a$ denotes the unique y such that $x \xrightarrow{a} y$ when exists. For $q \in Q$ and $u \in A^*$, $q \xrightarrow{u}$ and $q.u$ are defined inductively: $q \xrightarrow{\varepsilon}$ for every q, with $q.\varepsilon = q$, and $q \xrightarrow{a.u}$ *iff* $q \xrightarrow{a}$ & $q.a \xrightarrow{u}$, with $q.(a.u) = (q.a).u$.

The *language* $\mathcal{L}(\mathcal{A})$ of an automaton $\mathcal{A}$ is the set of words $\{m \in A^* / \ q_0 \xrightarrow{m}\}$ labelling its paths from the initial state. The equivalence by permutations $\sim$ is the equivalence on $\mathcal{L}(\mathcal{A})$ generated by those pairs $(uabv, ubav)$ in $\mathcal{L}(\mathcal{A}) \times \mathcal{L}(\mathcal{A})$ such that $a \| b$ in $\mathcal{A}$. Let $\leq$ be the prefix order, and $\lesssim \ = (\leq \cup \sim)^*$ be the prefix preorder modulo permutations. The equivalence by permutations coincides with the equivalence induced by the preorder ($m \sim n$ iff $m \lesssim n$ and $n \lesssim m$), thus the following definition makes sense.

Definition 6 (Domain of Configurations of a Trace Automaton) *The set of* finite configurations *of a trace automaton $\mathcal{A}$ is $\mathcal{L}(\mathcal{A})/\sim$, and the* domain of configurations *$D(\mathcal{A})$ of $\mathcal{A}$ is the ideal completion of the ordered set $(\mathcal{L}(\mathcal{A})/\sim \, , \, \lesssim / \sim)$.*

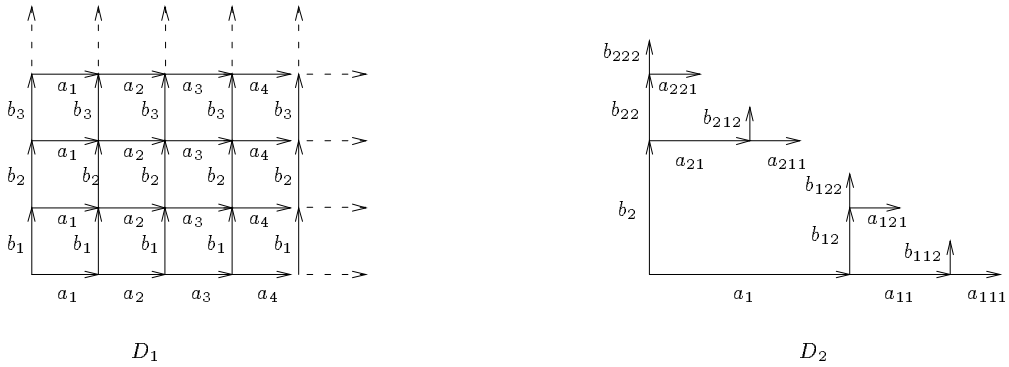


Figure 1: two recognizable event domains

Now $D \cong D(T_D)$ for any conflict event domain D. The explicit connection between conflict event domains and trace automata is stated by the following theorem.

Theorem 7 (E.W. Stark) *The domains of configurations of trace automata are the conflict event domains.*

Summing up theorems 3 and 7, we see that conflict event structures and trace automata are just alternative "syntactic" forms for event domains.

3 Recognizable Event Domains

This section presents the early steps of a general study aiming at an order theoretic characterization of the recognizable event domains defined as follows.

Definition 8 (Recognizable Event Domains) *A conflict event domain is* recognizable *if it is isomorphic to the domain of configurations of some finite trace automaton.*

Figure 1 shows two recognizable event domains $D_1 \cong D(\mathcal{A}_1)$ and $D_2 \cong D(\mathcal{A}_2)$, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are the two different trace automata with a single state q_0 and two transitions $q_0 \stackrel{a}{\rightarrow} q_0$ and $q_0 \stackrel{b}{\rightarrow} q_0$ which are obtained by setting $a \| b$ in $\mathcal{A}_1$ and $a \not\| b$ in $\mathcal{A}_2$. Given a (finite or infinite) trace automaton $\mathcal{A}$, and given

an event domain D inducing a trace automaton T_D as defined in section 2, the domain D is indeed isomorphic to the domain of configurations of the trace automaton $\mathcal{A}$ if and only if $\mathcal{A}$ is a *folding* of T_D according to the following definition, adapted from [BD93].

Definition 9 (Folding morphisms for Trace Automata) *A* folding morphism $(\eta, \sigma) : \mathcal{A}_1 \rightarrow \mathcal{A}_2$ *between trace automata is a pair of mappings* $\eta : A_1 \rightarrow A_2$, $\sigma : Q_1 \rightarrow Q_2$ *between their respective sets of actions and states, satisfying conditions 1 to 4 :*

1. $\sigma(q_{0,1}) = q_{0,2}$,
2. $q \xrightarrow{a} q'$ *in* T_1 *implies* $\sigma(q) \xrightarrow{\eta(a)} \sigma(q')$ *in* T_2,
3. *at every* $q \in Q_1$, η *restricts to a bijection between* $\{a / q \xrightarrow{a}\}$ *and* $\{b / \sigma(q) \xrightarrow{b}\}$,
4. *for every* $a, b \in A_1$, *if* $q \xrightarrow{a}$ *and* $q \xrightarrow{b}$ *in* $\mathcal{A}_1$ *for some* $q \in Q_1$ *then* $a \|_1 b \Leftrightarrow \eta(a) \|_2 \eta(b)$.

$\mathcal{A}_2$ *is said to be a* folding *of* $\mathcal{A}_1$ *(notation:* $\mathcal{A}_1 \geq \mathcal{A}_2$*) if there exists some folding morphism* (η, σ) *from* $\mathcal{A}_1$ *to* $\mathcal{A}_2$.

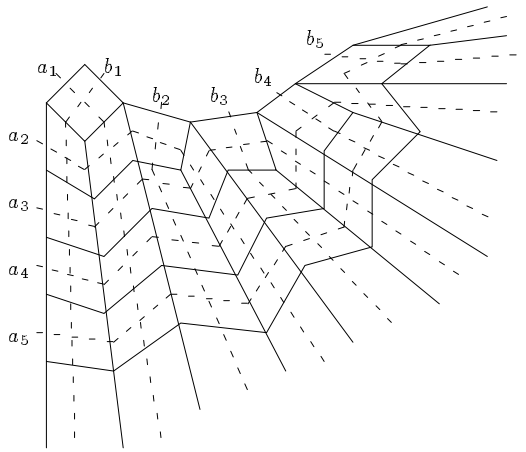
Theorem 10 (E. Badouel & Ph. Darondeau) *Let* $\mathcal{U}$ *be the* unfolding *operator that maps a trace automaton* $\mathcal{A}$ *to the acyclic trace automaton* $\mathcal{UA}{=}T_{D(\mathcal{A})}$ *derived from the domain of configurations of* $\mathcal{A}$, *then*

1. $\mathcal{UA} \geq \mathcal{A}$, *and*
2. $\mathcal{B} \geq \mathcal{A} \Rightarrow \mathcal{UA} \cong \mathcal{UB}$,

where $\cong$ *denotes isomorphism of trace automata.*

Corollary 11 *A conflict event domain* D *is recognizable if and only if the derived trace automaton* T_D *admits a finite folding* $T_D \geq \mathcal{A}$.

This corollary entails the non-recognizability of the event domain D shown in Fig. 2. As a matter of fact, any two events a_i and a_j such that $i \neq j$ are co-initial at some state and independent in T_D, hence from condition (4) in Def. 9, $\eta(a_i) \parallel \eta(a_j)$ and *a fortiori* $\eta(a_i) \neq \eta(a_j)$ for any folding morphism $(\eta, \sigma) : \mathcal{A} \rightarrow \mathcal{B}$, whereby $\mathcal{B}$ must be infinite. It is worth noting that the domain D has exactly five types of isomorphic types of *residuals*, where the residual D_x is the upper



$\emptyset \vdash a_1$

$\emptyset \vdash b_1$

$\{a_i\} \vdash a_{i+1}$

$\{b_i\} \vdash b_{i+1}$

$\{b_i\} \vdash a_{i+1}$

$a_i \ \# \ b_{i+1}$

Figure 2: a non recognizable event domain

restriction of D determined by x. We can now state two *necessary* conditions of recognizability for conflict event domains, inspired from the above, and propose a conjecture. First, a recognizable event domain D must have a finite number of types of isomorphic residuals, bounded by the number of states of any finite trace automaton $\mathcal{A}$ such that $D \cong D(\mathcal{A})$.

> This follows clearly from Def. 6 and the property of *left cancellation* (see [Sta89a]) stating that $uv \lesssim uw$ if and only if $v \lesssim w$ for the prefix ordering in the trace monoid over the concurrent alphabet of $\mathcal{A}$, which coincides when restricted to $\mathcal{L}(\mathcal{A})$ with the preorder $\lesssim$ of Def.6 (see again [Sta89a]).

Second, a recognizable event domain D must have bounded cliques of events for the relation $\natural$ defined as follows: $\natural$ is the least irreflexive and symmetric relation on E_D such that $e_1 \natural e_2$ if $e_1 \| _D e_2$ or $e_1 \#_D e_2$ or as in Fig.3 $e_1 \| _D e_3$ and $e_2 \#_D e_3$ for some event e_3 co-initial with e_1 and e_2 at two different states in T_D). If $D \cong D(\mathcal{A})$, and since condition (4) in Def. 9 entails $\eta(e_1) \| \eta(e_3)$ and $\eta(e_2) \not\| \eta(e_3)$ for e_1 , e_2 , and e_3 as above, the size of the $\natural$-cliques must actually be smaller than the size of the alphabet of $\mathcal{A}$.

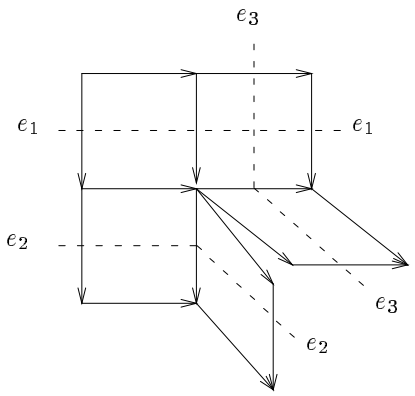


Figure 3: an example where $e_1 \natural e_2$: $e_1 \|_D e_3 \#_D e_2$

Conjecture 12 *A conflict event domain is recognizable if and only if it has simultaneously a finite number of non isomorphic residuals and bounded $\natural$-cliques.*

> This conjecture would be solved easily if any set E equipped with an irreflexive and symmetric relation $\natural$ could be decomposed into a finite partition $\{E_i | i \leq n\}$ made of subsets E_i *independent* of $\natural$, i.e. such that $E_i \times E_i \cap \natural = \emptyset$. If so, one could decompose further each subset E_i into a finite partition $\{E_{ij} | j \leq n_i\}$, such that $E_{ij} \times E_{i'j'} \cap \#_D = \emptyset$ or $E_{ij} \times E_{i'j'} \cap \|_D = \emptyset$ for every i, j and i', j'. Unfortunately, such decompositions do not exist for arbitrary relations $\natural$ (see ex.5 p.345 in [Ber73].

A detailed example in support of the above conjecture is presented at the end of the section. In the meantime, we restate the question of recognizability of conflict event domains in terms of finitely labelled dags. A folding morphism $(\eta, \sigma) : \mathcal{B} \to \mathcal{A}$ may always be split into a *labelling morphism* $(\eta, 1) : \mathcal{B} \to \mathcal{C}$, where $\mathcal{C}$ inherits the set of states of $\mathcal{B}$ and the concurrent alphabet of $\mathcal{A}$ and has transitions $q \stackrel{\eta(a)}{\to} q'$ reflecting transitions $q \stackrel{a}{\to} q'$ in $\mathcal{B}$, followed by a *state reduction* $(1, \sigma) : \mathcal{C} \to \mathcal{A}$. A labelling morphism $(\eta, 1) : \mathcal{B} \to \mathcal{C}$ may be interpreted as a *relabelling* of the trace automaton $\mathcal{B}$ according to the following definition.

Definition 13 (Relabelling of a Trace Automaton) *A* relabelling *of a trace automaton $\mathcal{A}$ is a mapping η from the alphabet of $\mathcal{A}$ onto a concurrent alphabet $(A, \|)$ such that, for every pair of actions a and b labelling transitions from some common state in $\mathcal{A}$, a and b are independent in $\mathcal{A}$ if and only if their images under η are independent in the new alphabet $(\eta(a)\|\eta(b))$.*

By splitting the folding morphism $(\eta, \sigma) : T_{D(\mathcal{A})} \to \mathcal{A}$, one obtains in particular a relabelling $(\eta, 1) : T_{D(\mathcal{A})} \to \mathcal{A}^*$ and a state reduction $(1, \sigma) : \mathcal{A}^* \to \mathcal{A}$ for $\mathcal{A}^*$ defined as follows.

Definition 14 (Labelled Unfolding of a Trace Automaton) *The* labelled unfolding *of a trace automaton $\mathcal{A}$ over the concurrent alphabet $(A, \|)$ is a trace automaton $\mathcal{A}^*$ over the same alphabet, with set of states $\mathcal{L}(\mathcal{A})/\sim$, initial state $\epsilon_\sim$ (the class of the empty word), and transitions $u_\sim \stackrel{a}{\to} (ua)_\sim$ for u and ua in $\mathcal{L}(\mathcal{A})$ and $a \in A$.*

When $\mathcal{A}$ is finite, and unlike the automaton $T_{D(\mathcal{A})}$ which may in general be constructed over an infinite alphabet, the acyclic automaton $\mathcal{A}^*$ has always finitely many *labelled residuals*, or directed subgraphs generated from an arbitrary element, up to isomorphism of labelled dags. Indeed, if we let $\mathcal{A}_q$ denote the modified version of $\mathcal{A}$ where q is taken as the initial state, every labelled residual of $\mathcal{A}^*$ is isomorphic to $\mathcal{A}^*_q$ for some state q in $\mathcal{A}$.

Observation 15 *A conflict event domain D is recognizable if and only if an acyclic trace automaton with finitely many non isomorphic labelled residuals may be obtained by relabelling the trace automaton T_D on a finite concurrent alphabet.*

> Indeed, if the trace automaton $\mathcal{A}$ is a relabelling of T_D with finitely many labelled residuals $\mathcal{A}_x$ (up to isomorphism) then $D \cong D(\mathcal{A}_*)$ for $\mathcal{A}_*$ defined on the set of (classes of isomorphic) residuals by setting $q \stackrel{a}{\to} q'$ in $\mathcal{A}_*$ if and only if there exists x and x' in D such that $\mathcal{A}_x \in q$, $\mathcal{A}_{x'} \in q'$ and $x \stackrel{a}{\to} x'$ in $\mathcal{A}$.

The rest of the section is devoted to the announced example.

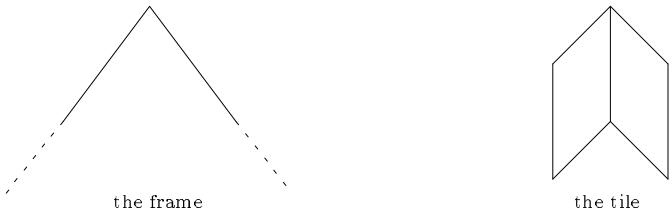


Figure 4: we pave the quaterplane ...

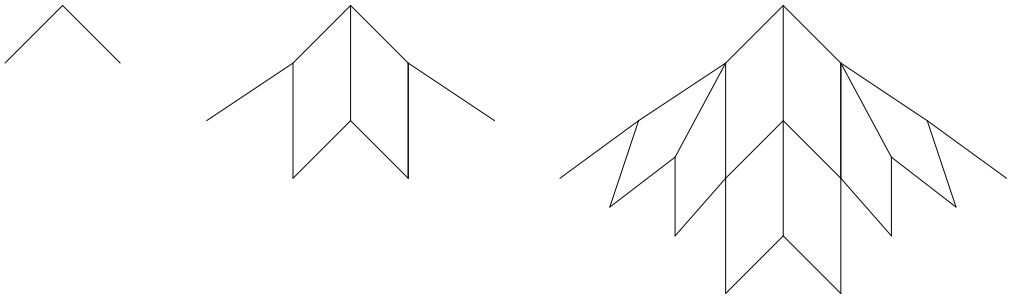


Figure 5: ... by inserting, at each stage, fresh copies of the tile in each free angle ...

Figure 4 shows a frame and a tile which may be used to pave that frame according to the sequence of steps indicated in Fig. 5 (at each step, fresh copies of the tile are inserted in each free angle). The tiling of the quaterplane so obtained yields the Hasse diagram H_D of a conflict event domain D, see Fig. 6. Domain D has exactly one type of (unlabelled) residuals, and its $\natural$ -cliques have size at most 8 (this bound is reached). As suggested by conjecture 12, this conflict event domain is recognizable. In view of observation 15, it suffices to relabel the trace automaton T_D associated with D on a finite concurrent alphabet, so as to obtain an acyclic trace automaton with finitely many types of isomorphic labelled residuals. The alphabet of T_D is the set of events represented with dashed lines in Fig. 6. This alphabet is reproduced in Fig. 7: each broken line represents a single event, and two intersecting lines represent two independent events. This relabelling problem may be solved on the concurrent alphabet $(\{1,\ldots,10\},\|)$, where the independence relation $\|$

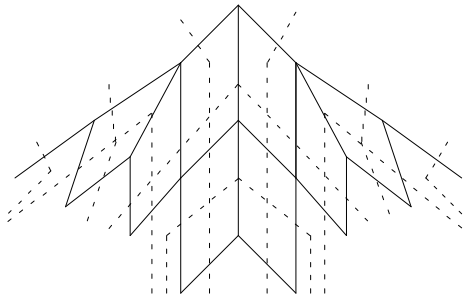


Figure 6: ...and we obtain the Hasse diagram of a conflict event domain.

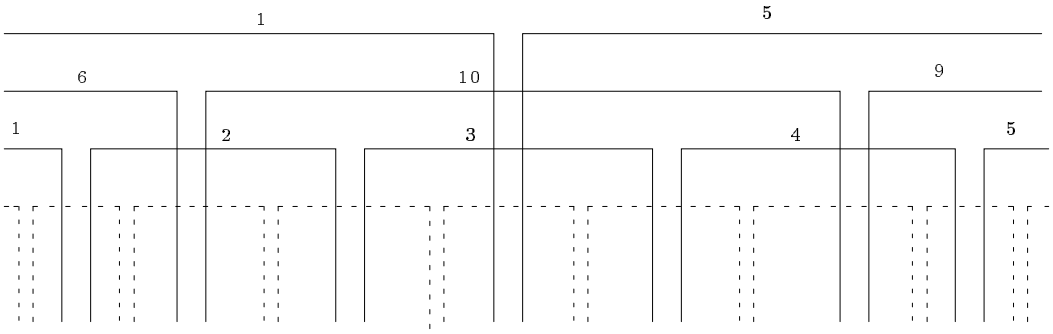


Figure 7: the events of the conflict event domain of Fig. 6

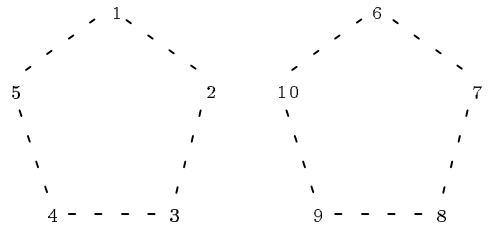


Figure 8: a concurrent alphabet

is the complement of the conflict relation # visualized by the two rings in Fig. 8. More precisely, $\|$ is the complement of # minus the identity relation. The events are labelled from left to right on Fig. 7, using at each level one of the two rings taken in alternance. Numbers are chosen in their order on the rings from the respective origins 1 and 6, but the rotation is reversed every two levels. The resulting sequence of labels for the events of Fig. 7 is the following.

1 , 5
6 , 10 , 9
1 , 2 , 3 , 4 , 5
6 , 7 , 8 , 9 , 10 , 6 , 7 , 8 , 9
1 , 5 , 4 , 3 , 2 , 1 , 5 , 4 , 3 , 2 , 1 , 5 , 4 , 3 , 2 , 1 , 5
.....

The reader may verify that the labelled version of T_D has indeed 20 types of isomorphic labelled residuals.

The relative intricacy of the solution to the relabelling problem for this simple example is typical of the non context-free event domains, which may be characterized (see appendix 6.1) by the presence of arbitrary long elementary zig-zags for the covering relation on finite elements.

Definition 16 (Zig-zags) *A* zig-zag *for a relation $\rightarrow \subset X \times X$ is a sequence $(x_i)_{i\in I}$ of elements of X,indexed by an interval $I \subset \mathbb{N}$, such that $x_{2n} \rightarrow x_{2n-1}$ and $x_{2n} \rightarrow x_{2n+1}$ whenever the corresponding indices are in I. In particular, it is a path $[x_i, x_{i+1}, \ldots, x_{i+k}]$ of the undirected graph $\mathbf{G} = (X, -)$ where $- = \rightarrow \cup \rightarrow^{-1}$. A path is said to be elementary when it does not pass twice by the same vertex, thus a zig-zag is* elementary *if all the elements x_i are distinct.*

We give in the next section a full solution to the relabelling problem for the context-free event domains, where elementary zig-zags for $\prec$) are bounded.

4 Context-Free Event Domains

A context-free graph [MS85] is a rooted graph of finite degree such that, by removing all vertices within some arbitrary distance from the root, one obtains a bounded number of types of isomorphic connected components. This characteristic property determines exactly the class of rooted graphs of finite degree which may be generated from deterministic graph grammars [Cau92]. In this section, we focus our attention on context-free event domains, defined as follows.

Notation 17 *For any domain D, let H_D denote the Hasse diagram or transitive reduction of the order in D, restricted to the finite elements of D.*

Definition 18 (Context-Free Event Domains) *A conflict event domain D is* context-free *if H_D is a context-free graph.*

The purpose of this section is to provide a constructive proof for the following theorem.

Theorem 19 *Context-free event domains are recognizable.*

Thus, we will construct from a graph grammar for H_D a finite trace automaton $\mathcal{A}$ such that $D(\mathcal{A}) \cong D$. The running example for the section is the context-free event domain whose Hasse diagram (generated by a graph grammar with a single replacement rule) is sketched in Fig. 9. That dag may be seen as a thick binary tree whose branches are assembled from squares. Since zig-zags have length at most 6, the picture is "full of gaps".

4.1 Graph Grammars Generating Event Domains

In this subsection, we show that any *monotone* graph grammar which generates the Hasse diagram of an event domain may be normalized to a monotone and *uniform* graph grammar, yielding a natural encoding of vertices and edges

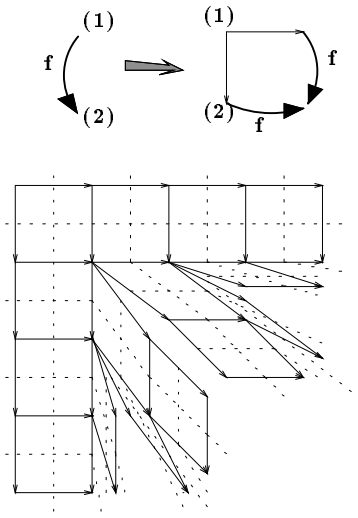


Figure 9: a context-free event domain

by words. The projective order on prime intervals is then encoded by a finite suffix-rewrite system on words. The various encodings are exploited in a second subsection, where trace automata are derived from monotone and uniform graph grammars.

Definition 20 (Graph Grammar) *Let* $\Sigma = \{\Sigma_n | n \geq 1\}$ *be an alphabet with arities and* V *be a set of vertices. A* hyperarc *is any word in* $\bigcup_{n\geq 1} \Sigma_n \cdot V^n$. *The hyperarc* $fx_1 \dots x_n$ *is* labelled *by* f, *and it connects in that order the vertices* $x_1 \dots x_n$. *A* hypergraph H *on* (Σ, V) *is a graph* $[H] = (V, R)$ *on* V *(hence* $R \subseteq V \times V$*) called the* underlying graph *of* H, *together with a set of hyperarcs* $fx_1 \dots x_n$ *connecting vertices in* V. *A* hyperarc replacement rule *is a production* $fx_1 \dots x_n \rightarrow H_f$ *where* $fx_1 \dots x_n$ *is a hyperarc,* H_f *is a* finite *hypergraph on* (Σ, V) *and* $\{x_1, \dots, x_n\}$ *are* distinct *vertices of* H_f. *A* graph grammar *on* Σ *is a finite set of hyperarc replacement rules* $fx_1 \dots x_n \rightarrow H_f$ *where* $f \in \Sigma_n$.

A graph grammar G is said to be *deterministic* if every symbol $f \in \Sigma_n$ occurs exactly once in the left member of a rule $fx_1 \dots x_n \rightarrow H_f$. A deterministic graph grammar G defines a vector of graphs $\{G_f^\omega | f \in \Sigma\}$ obtained by infinite iterations of the hyperarc replacement rules $fx_1 \dots x_n \rightarrow H_f$ from the respective hyperarcs

$fx_1 \ldots x_n$. Provided we add enough copies of rules, we may assume w.l.o.g. that a non terminal symbol g has at most one occurrence in the right hand side of each rule $fx_1 \ldots x_n \to H_f$, in which case we set $g \in \mathbf{succ}(f)$. Under this assumption, the graph $G_f^\omega = (V_f^\omega, R_f^\omega)$ may be described as follows. Let $\mathcal{T}$ denote the set of Σ-words $f_1 \ldots f_n$ where $\forall i < n \ f_{i+1} \in \mathbf{succ}(f_i)$. Let $\mathcal{T}_f$ (respectively $\mathcal{T}^g$) be the subset of words $f_1 \ldots f_n$ in $\mathcal{T}$ such that $f_1 = f$ (resp. $f_n = g$), and $\mathcal{T}_f^g = \mathcal{T}_f \cap \mathcal{T}^g$. The set $\mathcal{T}_f$, ordered by the prefix ordering, may be seen as a deterministic tree, called the *parse-tree* of G_f^ω. Finally, let $[H_f] = (V_f, R_f)$ be the underlying graph of H_f. Then $V_f^\omega = \{< u, x >_\equiv \ / \ u \in \mathcal{T}_f^g \ \wedge \ x \in V_g\}$ where $\equiv$ is the least equivalence on $\mathcal{T} \times V$ such that $< uhg, x_i > \equiv < uh, y_i >$ for $1 \leq i \leq n$ whenever $gy_1 \ldots y_n$ is a hyperarc of H_h and $gx_1 \ldots x_n \to H_g$ is the replacement rule for g; and $(X, Y) \in R_f^\omega$ if, and only if, $< u, x > \in X$ and $< u, y > \in Y$ for some x, y and $u \in \mathcal{T}_f^g$ such that $(x, y) \in R_g$. Extending the notation, we let $G^\omega(M)$ denote the graph generated from $f_0 x_1$ by the deterministic graph grammar with set of rules $\{f_0 x_1 \to M\} \cup G$. According to [Cau92], context-free graphs may be defined as follow.

Definition 21 (Context-Free Graph) *A* context-free *graph is a rooted graph of finite degree which is isomorphic to $G^\omega(M)$ for some deterministic graph grammar G and finite hypergraph M.*

Definition 22 (Monotone Graph Grammar) *A graph grammar G is* monotone *if the following four conditions are satisfied by every hyperarc replacement rule $fx_1 \ldots x_n \to H$:*

1. *the underlying graph of H is a directed acyclic graph (or dag),*
2. *the vertices $x_1, \ldots, x_n$ are the minimal elements of $[H]$, called* input vertices *of H,*
3. *the vertices that are connected by hyperarcs $gy_1 \ldots y_m$ in H are maximal elements of $[H]$, called* ouput vertices *of H,*
4. *the input and ouput vertices of H are disjoint.*

Proposition 23 *An a-transitive and rooted dag of finite degree is context-free if, and only if, it is isomorphic to G_f^ω for some deterministic and monotone grammar G and for some unary symbol f.*

Proof: Let D be an a-transitive and rooted dag of finite degree. Assume D is context-free. Thus, by removing all vertices within some arbitrary distance from the root, one obtains finitely many types of isomorphic connected components (according to Müller and Schupp's characterization of context-free graphs). Let us fix a representing dag D_h for each type h, and let $x_1 \dots x_{a(h)}$ be an enumeration of the minimal vertices of D_h. We construct a deterministic graph grammar G on (Σ, V), where Σ is the alphabet of types h with arities $a(h)$, and V is the union of subsets of vertices located at depth 0 or 1 in dags D_h. For each h in Σ, we set a single replacement rule $hx_1 \dots x_{a(h)} \to H_h$, where H_h is a finite hypergraph of depth 1 or 0 (in which case $a(h) = 1$ and H_h is reduced to the isolated vertex x_1). The hypergraph H_h is derived from D_h in two stages. In a first stage, every ordered sequence of vertices $y_1 \dots y_n$ left as the minimal vertices of some connected component after removing $x_1 \dots x_{a(h)}$ is equipped with a hyperarc $gy_1 \dots y_n$, where g is the type of that component. In a second stage, all the vertices located at depth strictly greater than 1 are removed. The resulting grammar G is deterministic and monotone, and $D \cong G^\omega_f$ where f is the isomorphism type of D (whence $a(f) = 1$ since D is rooted). ∎

It may be observed that the monotone graph grammar constructed in the proof of the above proposition is also uniform according to the following definition, adapted from [Cau92].

Definition 24 (Uniform Graph Grammar) *A deterministic graph grammar G over Σ is* connected *if all the graphs G^ω_f ($f \in \Sigma$) are connected. A deterministic and connected graph grammar G is* uniform *if the following three conditions are satisfied in every hypergraph H_f such that $fx_1 \dots x_n \to H_f$ in G, except in the possible case where H_f reduces to a single vertex:*

1. *all vertices of hyperarcs are shared with arcs, whereas one vertex can neither occur twice on a hyperarc nor be shared by two hyperarcs,*
2. *all vertices in the set $\{x_1 \dots x_n\}$ occur on arcs and they do not occur on hyperarcs,*
3. *at least one of the vertices $x_1 \dots x_n$ occurs on each arc.*

Theorem 25 (D. Caucal) *Any pair (Γ, M) made of a deterministic graph grammar Γ and of a finite hypergraph M such that $\Gamma^\omega(M)$ is a connected and non empty graph of finite degree may be transformed into a uniform grammar*

G such that $G^\omega(M)$ and $\Gamma^\omega(M)$ are isomorphic. Moreover, G does not depend on M.

Observation 26 *If Γ is monotone then the uniform grammar G is also monotone.*

Theorem 27 *Given a monotone and uniform graph grammar G over Σ and a unary hyperarc symbol $f_0 \in \Sigma$, one can decide whether the context-free graph $G^\omega_{f_0} = G^\omega(f_0 x_1)$ coincides with the Hasse diagram of the order on finite elements in some conflict free event domain.*

Proof: see Appendix 6.2.
Suppose that an event domain D has been specified by a deterministic and monotone graph grammar G, generating the Hasse diagram of D from the axiom $f_0 x_1$ (i.e. $H_D \cong G^\omega_{f_0}$). In view of Theo. 25 and the above observation, one can always transform G into a uniform graph grammar. The benefit of the transformation is to yield a syntactic presentation of H_D, described hereafter. From now on, we assume that $G = \{f x_1 \dots x_n \to H_f\}_{f \in \Sigma}$ is both monotone and uniform , and we carry on assuming that every $g \in \Sigma$ occurs at most once on the right hand side of each rule. Thus, unless H_f reduces to a single vertex, its underlying graph $[H_f]$ is the Hasse diagram of a partial order of depth 1 with $a(f)$ minimal elements, and whose set of maximal elements is partitioned by hyperarcs. Each replacement rule $f x_1 \dots x_{a(f)} \to H_f$ may now be encoded by a corresponding graph $G_f = (V_f, A_f)$, isomorphic to $[H_f]$. The set of vertices $V_f = I_f \cup O_f$ is divided into the *input vertices* $I_f = \{(f,1), \dots, (f, a(f))\}$ and the *output vertices* $O_f = \{(fg, i) | g \in \mathbf{succ}(f) \text{ and } 1 \le i \le a(g)\}$. The set of arcs $A_f \subseteq I_f \times O_f$ is the set of pairs $((f,i),(fg,j))$ such that there is an arc from x_i to the j^{th} vertex of hyperarc g in H_f. Thus, in particular, $V_f = \{(f,1)\}$ and $A_f = \emptyset$ if the replacement rule for f is $f x_1 \to x_1$. Figure 11 displays the graphs which encode the rules of the grammar shown in Fig. 10.

Proposition 28 *If the grammar G is monotone and uniform then $G^\omega_f \cong \bigcup \{G_v | v \in \mathcal{T}_f\}$, where $G_v = u \cdot G_g = (u \cdot V_g, u \cdot A_g)$ for every $v = ug \in \mathcal{T}_f$ with $u \cdot (w,i) = (uw,i)$ and $u \cdot ((w,i),(w',i')) = ((uw,i),(uw',i'))$, and the union of graphs is defined componentwise, i.e. $\bigcup_{i \in I}(V_i, A_i) = (\bigcup_{i \in I} V_i, \bigcup_{i \in I} A_i)$.*

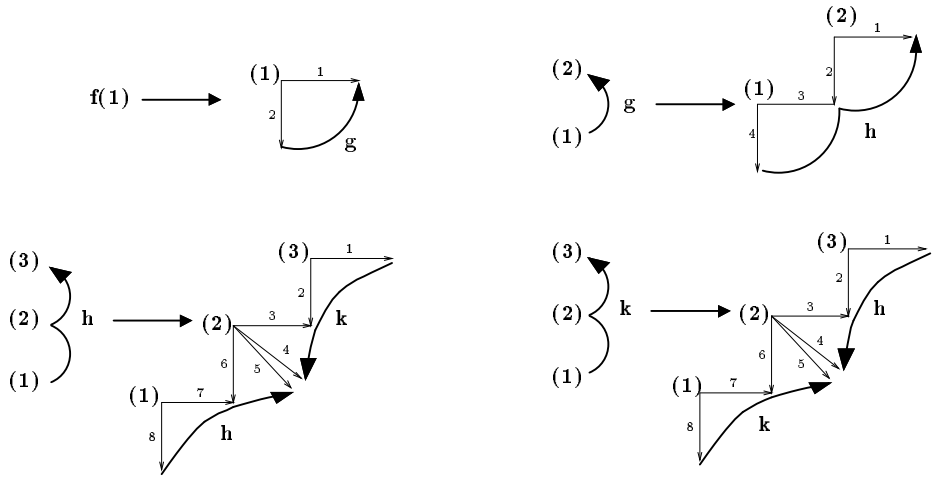


Figure 10: a uniform graph grammar generating the context-free event domain of Fig. 9

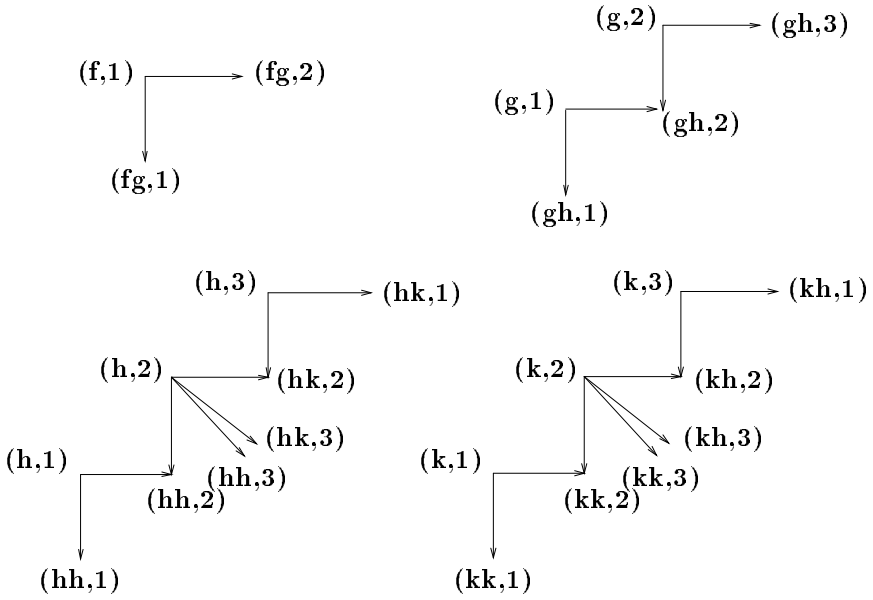


Figure 11: graphs G_f coding the production rules $f \rightarrow H_f$ of Fig. 10

The above proposition suggests simple notations for the vertices and arcs of H_D $(= G^\omega_{f_0})$. Let us adopt for $G^\omega_f = (V^\omega_f, A^\omega_f)$ the variant definition $G^\omega_f = \bigcup\{G_v | v \in \mathcal{T}_f\}$. Then the vertices in V^ω_f are mapped bijectively to pairs (u, i) such that $u \in \mathcal{T}^g_f$ and $1 \leq i \leq a(g)$, and if we fix for each $f \in \Sigma$ an enumeration of the arcs in H_f (as in Fig. 10), the arcs in A^ω_f are mapped bijectively to words ui such that $u \in \mathcal{T}^g_f$ and $1 \leq i \leq n(g)$, where $n(g)$ is the number of arcs in A_g (see Fig. 12). Before we tackle the encoding of the projective order on prime intervals in H_D, let us introduce a few definitions used in the second subsection.

Definition 29 (Sections) *For $v = u.g \in \mathcal{T}_f$, let $u : G^\omega_g \hookrightarrow G^\omega_f$ be the graph embedding $u \cdot (V^\omega_g, A^\omega_g) = (u \cdot V^\omega_g, u \cdot A^\omega_g)$; then $G^\omega_f > v = u \cdot G^\omega_g$ is the* residual *of G^ω_f by v. The set $\varphi(u) = \{(u,1), \ldots, (u,n)\}$ of minimal vertices of $G^\omega_f > u$ is called the* section *(of G^ω_f) determined by $u \in \mathcal{T}_f$ - thus, the set of sections of G^ω_f defines a partition of V^ω_f.*

In section 2, events have been defined as equivalence classes of projective prime intervals in Scott domains, and they have been equipped with two disjoint binary relations, called independence and conflict. Let us extend these various definitions from event domains D to context-free graphs $G^\omega_{f_0}$, in such a way that the new definitions agree with the earlier ones for $G^\omega_{f_0} \cong H_D$.

Definition 30 (Projective Order and Events in Context Free Dags) *Let $\mathbf{G} = (\mathbf{V}, \prec)$ be the transitive reduction of an acyclic graph. A* prime interval *$[x, y]$ is a pair of vertices such that $x \prec y$. A prime interval $[x, y]$ is* covered *by a prime interval $[x', y']$ (notation $[x, y] \prec [x', y']$) if $x \prec x'$ and $y \prec y'$, which gives rise to a* diamond *$\diamond(x, x', y, y')$. The* projective order *$\sqsubseteq$ on prime intervals is the reflexive and transitive closure of $\prec$. An* event *is an equivalence class of prime intervals for the equivalence generated by $\sqsubseteq$ (thus, two prime intervals are equivalent if and only if they are connected by a zig-zag in the projective order). Let $\bigvee(x, y, z)$ when $x \prec y$, $x \prec z$ but there is no t such that $\diamond(x, y, z, t)$, then two events e_1 and e_2 are in* conflict *(notation $e_1 \# e_2$) if $\bigvee(x, y, z)$ for some $[x, y] \in e_1$ and $[x, z] \in e_2$, and they are* independent *(notation $e_1 \| e_2$) if $\diamond(x, y, z, t)$ for some t such that $[x, y] \in e_1$ and $[x, z] \in e_2$.*

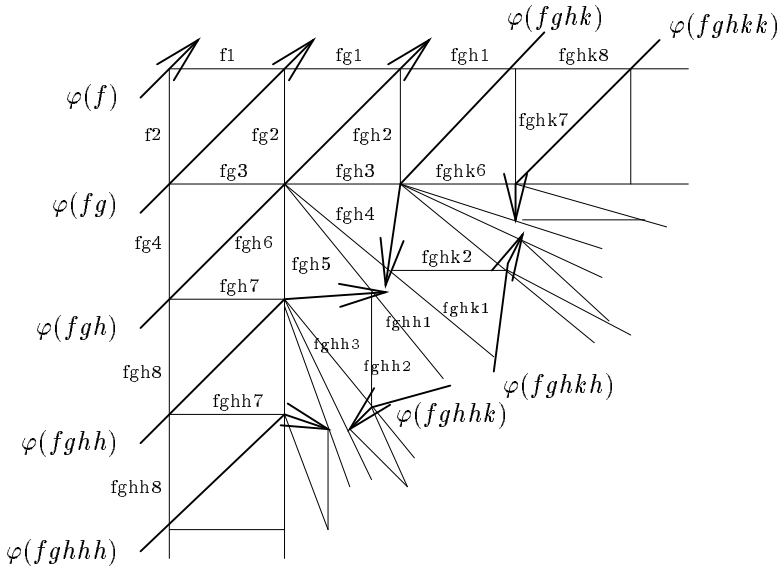


Figure 12: sections in the domain of Fig. 9 induced by the graph grammar of Fig. 10

For instance, in Fig. 12, the prime intervals $fg1 \sqsubseteq fgh3 \sqsubseteq fghk2$ are linearly ordered and belong to the same event. Now, with our encoding of prime intervals, the projective order may be represented by a finite suffix rewrite system constructed from the graph grammar. In order to see that, observe that whenever $\diamond(x, x', y, y')$ in G^ω_f, x belongs to a section $\varphi(ug)$ such that, for some $h \in \mathbf{succ}(g)$, $[x, y] = ugi$, $[x, x'] = ugj$, $[y, y'] = ughj'$, and $[x', y'] = ughi'$ for some $i, j \in \{1, \ldots, n(g)\}$ and $i', j' \in \{1, \ldots, n(h)\}$ (see Fig. **??**), whence the following.

Observation 31 *$wi \prec w'i'$ if and only if $w = u \cdot g$, $w' = u \cdot gh$ and $gi \prec ghi'$.*

Observation 32 *$ufi \sqsubseteq vgj$ if and only if $v = u \cdot w$ and $fi \xrightarrow{*} wgj$, where $\xrightarrow{*}$ is the derivation relation in the suffix rewrite system with rules $gi \to ghi'$ such that $g, h \in \Sigma$ and $gi \prec ghi'$ in G^ω_g (or in its approximation at depth 2, see Fig. 13).*

Notation 33 *For $u \in \mathcal{T}^g_f$ and $1 \leq i \leq n(g)$, let $u(i)$ denote the event of G^ω_f which contains the prime interval ui.*

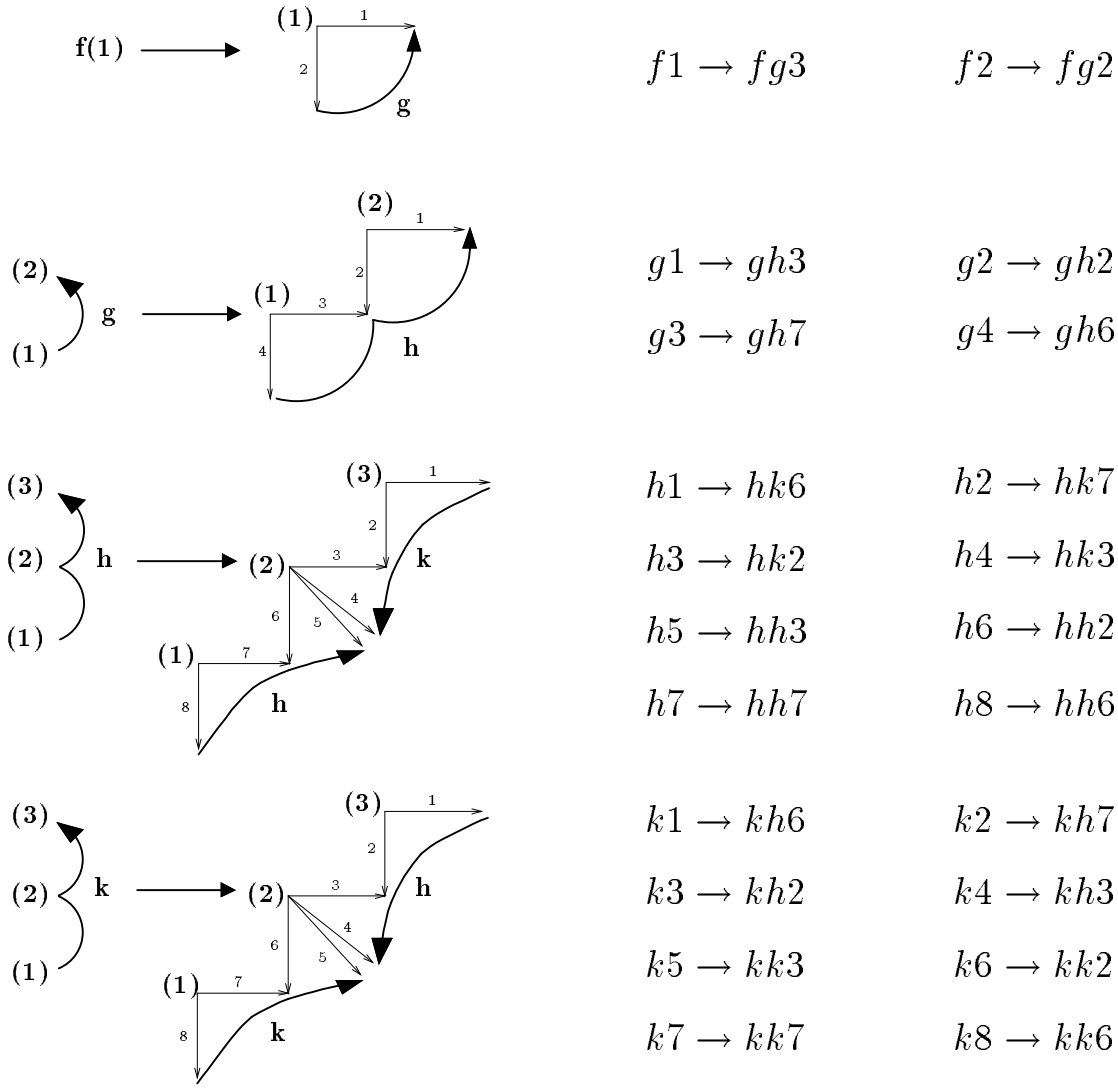
For instance, in Fig. 12, $fg(1) = fgh(3) = fghk(2)$. Thus each word $u \in \mathcal{T}^g_f$ denotes a corresponding mapping from $dom(u) = \{1, \ldots, n(g)\}$ to the set of events of G^ω_f; let $Im(u)$ denote the image of this mapping.

4.2 Extracting a Trace Automaton from a Graph Grammar

In this subsection we exploit the syntactic presentation of the context-free domain D induced by the uniform and monotone grammar G in order to derive a finite trace automaton $\mathcal{A}$ whose domain of configuration is isomorphic to D. In a first step, we cut down $H_D \cong G^\omega_{f_0}$ to a representative subgraph of finite size, delimited by *repeated sections*.

Definition 34 (Repeated Sections) *Two* distinct *Σ-words $u, v \in \mathcal{T}_{f_0}$ induce repeated sections (notation: $u \ll v$) if the following conditions hold :*

1. *u is a prefix of v: $u < v$,*
2. *u and v end with the same symbol: $u = u' \cdot f$ and $v = v' \cdot f$,*
3. *$\forall i, j \in \{1, \ldots, n(f)\}$ $u(i) = u(j) \Leftrightarrow v(i) = v(j)$, and $u(i) \|_D u(j) \Leftrightarrow v(i) \|_D v(j)$.*



$$f1 \to fg3 \qquad f2 \to fg2$$

$$g1 \to gh3 \qquad g2 \to gh2$$
$$g3 \to gh7 \qquad g4 \to gh6$$

$$h1 \to hk6 \qquad h2 \to hk7$$
$$h3 \to hk2 \qquad h4 \to hk3$$
$$h5 \to hh3 \qquad h6 \to hh2$$
$$h7 \to hh7 \qquad h8 \to hh6$$

$$k1 \to kh6 \qquad k2 \to kh7$$
$$k3 \to kh2 \qquad k4 \to kh3$$
$$k5 \to kk3 \qquad k6 \to kk2$$
$$k7 \to kk7 \qquad k8 \to kk6$$

Figure 13: a uniform graph gammar G and the suffix rewrite system generating the projective order in G_f^ω

Let $\mathcal{T}_\Delta$ be the initial subtree of the parse tree $\mathcal{T}_{f_0}$ which remains after removing all nodes $w \in \Sigma^*$ such that $u < v < w$ for some pair of repeated sections $u \ll v$ in H_D. We say that two words $u, v \in \mathcal{T}_{f_0}$ have the same *type* when the last two conditions of Def. 34 are met. By Koenig's lemma, $\mathcal{T}_\Delta$ has no infinite branch since there are only finitely many types. A uniform procedure constructing $\mathcal{T}_\Delta$ from G and f_0 stems immediately from the next lemma, established in appendix 6.2.

Lemma 35 *Given $u \in \mathcal{T}_{f_0}$ and $i, j \in dom(u)$, one may decide from G and f_0 whether $u(i) = u(j)$, and whether $u(i) \|_D u(j)$, where $D \cong G^\omega_{f_0}$.*

Let $\mathbf{Fr}(\mathcal{T}_\Delta)$ denote the *frontier* of $\mathcal{T}_\Delta$, i.e. the set of maximal words of $\mathcal{T}_\Delta$, and let $\mathcal{T}^\circ_\Delta = \mathcal{T}_\Delta \backslash \mathbf{Fr}(\mathcal{T}_\Delta)$ denote the *interior* of $\mathcal{T}_\Delta$. Let $\{\beta_1, \ldots, \beta_n\}$ be an enumeration of $\mathbf{Fr}(\mathcal{T}_\Delta)$, and for $1 \leq i \leq n$ let α_i be the unique word in $\mathcal{T}^\circ_\Delta$ such that $\alpha_i \ll \beta_i$. Clearly, the frontier of $\mathcal{T}_\Delta$ is a *cut* of $\mathcal{T}_{f_0}$ (i.e. a maximal subset of pairwise incomparable elements). Moreover, since α_i and β_i have the same ending symbol, the following holds by definition of $\mathcal{T}_{f_0}$:

$$\forall w \in \Sigma^* \quad \alpha_i \cdot w \in \mathcal{T} \Leftrightarrow \beta_i \cdot w \in \mathcal{T}$$

Thus we can reduce any word $u \in \mathcal{T}_{f_0}$ by the prefix rewriting system $\Delta = \{\beta_i \xrightarrow{i} \alpha_i | \alpha_i \ll \beta_i\}$: this rewriting system is deterministic (because the words β_i are incomparable) and noetherian (because the length of α_i is strictly less than the length of β_i), hence any word $u \in \mathcal{T}_{f_0}$ reduces to a unique normal form $u' = u \ \mathbf{mod}\ \Delta$ and clearly $u' \in \mathcal{T}^\circ_\Delta$.

We can now clarify the role of the last condition stated for u and v in the definition of repeated sections. First of all, this condition entails that composite relations $u^{-1} \cdot v$ and $v^{-1} \cdot u$ (where u and v denote functions on events according to notation 33) are functional relations on E_D, inducing reciprocal bijections $\psi^v_u : Im(u) \to Im(v)$ and $\psi^u_v : Im(v) \to Im(u)$ such that:

$$\forall i \ \ 1 \leq i \leq n(f) \quad \psi^v_u(u(i)) = v(i) \tag{1}$$

More importantly, the mapping ψ^v_u is a *partial isomorphism* of the graph $(E_D, \|_D)$, with domain $Im(u)$ and range $Im(v)$.

Definition 36 (Partial Isomorphism) *A partial isomorphism of a relational structure G is an isomorphism $\psi : D \to R$ between two substructures of G called respectively the domain and the range of ψ.*

We recall that H is a *substructure* of G if H has the structure induced by G or equivalently if G is a *conservative extension* of H ($R(x_1,\ldots,x_n) \in G$ and $\forall i \cdot x_i \in H$ entail $R(x_1,\ldots,x_n) \in H$).

Thus, the repeated sections $\alpha_i \ll \beta_i$ $(1 \leq i \leq n)$ determine a family $\{\psi_1,\ldots,\psi_n\}$ of partial isomorphisms $\psi_i = \psi_{\alpha_i}^{\beta_i}$ of the concurrent alphabet $(E_\Delta, \|_\Delta)$, where E_Δ is the set of events occurring in the representative part of H_D, i.e. $E_\Delta = \bigcup\{Im(u)/\ u \in \mathcal{T}_\Delta\}$, and $\|_\Delta$ is the restriction of $\|_D$ to E_Δ. The following result, which is the crux of our construction, was first established by Hrushovski in [Hru92].

Theorem 37 (E. Hrushovski) *A finite graph $(E_\Delta, \|_\Delta)$ equipped with a finite family of partial isomorphisms $\{\psi_1,\ldots,\psi_n\}$ can always be extended conservatively into a finite graph $(E, \|)$ equipped with automorphisms $\{\Psi_1,\ldots,\Psi_n\}$ extending the former ones.*

Hrushovski constructs in fact automorphisms Ψ_i which do not depend on relation $\|_\Delta$ but only on the partial isomorphisms. His construction was extended to relational structures in [Her94]. A simpler proof of Hrushovski's theorem is due to Lascar, who constructs in [Las94] a graph $(E, \|)$ in which E does not depend upon the partial isomorphisms ψ_i but only on the graph $(E, \|)$.

The above theorem indicates that one can always extend $(E_\Delta, \|_\Delta)$ into a finite alphabet $(E, \|)$ large enough to permit constructing from D a trace automaton with underlying graph H_D, by labelling the prime intervals of D according to a recursive procedure defined by the automorphisms Ψ_i and the reduction rules $\beta_i \xrightarrow{i} \alpha_i$ in Δ. Moreover, the possibly infinite trace automaton obtained in this way may always be folded to a finite and equivalent trace automaton.

Definition 38 (Labels of Prime Intervals) *Let $\Psi_1,\ldots,\Psi_n$ be automorphisms of $(E, \|)$ constructed according to Theo. 37 from $(E_\Delta, \|_\Delta)$ and the partial isomorphisms ψ_i induced by the repeated sections $\alpha_i \ll \beta_i$ $(1 \leq i \leq n)$ of H_D. Then each prime interval $u \cdot j$ of domain D is given a* label $l_\Delta(u \cdot j) \in E$, *defined as follows:*

1. $l_\Delta(u \cdot j) = u(j) \in E_\Delta$ *if* $u \in \mathcal{T}_\Delta^\circ$,
2. $l_\Delta(u \cdot j) = \Psi_i(l_\Delta(v \cdot j))$ *if* $u \xrightarrow{i} v$ *(in* Δ*).*

We proceed below to a series of verifications, necessary for proving that the labelled version of H_D is a trace automaton.

Observation 39 *The label of a prime interval $u \cdot j$ such that $u \in \mathcal{T}_\Delta$ is given by $l_\Delta(u \cdot j) = u(j)$.*

Actually, this identity holds by definition of l_Δ for $u \in \mathcal{T}_\Delta^\circ$; and for $u = \beta_i$ we have $l_\Delta(u \cdot j) = l_\Delta(\beta_i \cdot j) = \Psi_i(l_\Delta(\alpha_i \cdot j))$ [because $\beta_i \xrightarrow{i} \alpha_i$] $= \Psi_i(\alpha_i(j))$ [because $\alpha_i \in \mathcal{T}_\Delta^\circ$] $= \beta_i(j)$ [because Ψ_i is the extension of $\psi_i = \psi_{\alpha_i}^{\beta_i}$] $= u(j)$. ∎

Lemma 40 *Two prime intervals which belong to the same event have the same label.*

Proof: We show $u \cdot i \mathrel{-\!\!\prec} v \cdot j \;\Rightarrow\; l_\Delta(u \cdot i) = l_\Delta(v \cdot j)$ by induction on the length of the normalisation process for $u \cdot i$ (in the prefix rewrite system Δ).

1. **Base case:** $u \in \mathcal{T}_\Delta^\circ$
   Then $v \in \mathcal{T}_\Delta$ and by the above observation $l_\Delta(u \cdot i) = u(i) = v(j)$ [because $u \cdot i \mathrel{-\!\!\prec} v \cdot j$] $= l_\Delta(v \cdot j)$.

2. **General Case:** $u \notin \mathcal{T}_\Delta^\circ$
   Then $u \xrightarrow{k} u'$ where $u = \beta_k w f$ and $u' = \alpha_k w f$. Since $u \cdot i \mathrel{-\!\!\prec} v \cdot j$, $v = \beta_k w f g$ for some g such that $f \cdot i \to fg \cdot j$ in the suffix rewrite system which encodes the projective order (see Obs. 39). Thus $v \xrightarrow{k} v' = \alpha_k w f g$ (using prefix rewriting) and $u' \cdot i \mathrel{-\!\!\prec} v' \cdot j$ (using suffix rewriting). By the inductive assumption, $l_\Delta(u' \cdot i) = l_\Delta(v' \cdot j)$, hence $l_\Delta(u \cdot i) = \Psi_k(l_\Delta(u' \cdot i)) = \Psi_k(l_\Delta(v' \cdot j)) = l_\Delta(v \cdot j)$.

∎

Definition 41 *Let $\eta_\Delta : E_D \to E$ be the labelling function on events induced from mapping l_Δ.*

Lemma 42 *Two independent events are always labelled by independent letters in the concurrent alphabet $(E, \|)$.*

Proof: Let $e_1, e_2 \in E_D$ such that $e_1 \| _D e_2$ then $e_1 = ug(i)$ and $e_2 = ug(j)$ for some diamond based on ugi and ugj (see Fig. **??**). We show $\eta_\Delta(e_1) \| \eta_\Delta(e_2)$ by induction on the length of the normalisation process for $u \cdot g$.

1. **Base case:** $ug \in \mathcal{T}_\Delta^\circ$
   $\eta_\Delta(e_1) = l_\Delta(ug \cdot i) = ug(i) \|_\Delta ug(j) = l_\Delta(ug \cdot j) = \eta_\Delta(e_2)$ and $\eta_\Delta(e_1) \| \eta_\Delta(e_2)$ because $(E_\Delta, \|_\Delta)$ is a restriction of $(E, \|)$.

2. **General Case:** $ug \notin \mathcal{T}_\Delta^\circ$
   Then $ug \xrightarrow{l} u'g$ where $ug = \beta_l wg$ and there exists a diamond based on $u'gi$ and $u'gj$. By the inductive hypothesis, $l_\Delta(\alpha_l wg \cdot i) \| l_\Delta(\alpha_l wg \cdot j)$, hence $\eta_\Delta(e_1) = l_\Delta(ug \cdot i) = \Psi_l(l_\Delta(\alpha_l wg \cdot i)) \parallel \Psi_l(l_\Delta(\alpha_l wg \cdot j)) = l_\Delta(ug \cdot j) = \eta_\Delta(e_2)$ because Ψ_l is an automorphism of $(E, \|)$.

∎

Lemma 43 *Two conflicting events are never labelled by independent letters in the concurrent alphabet* $(E, \|)$.

Proof: Let $e_1, e_2 \in E_D$ such that $e_1 \#_D e_2$ then $\bigvee(x, y, z)$ in H_D for some x, y, z such that $[x, y] = ugi$, $[x, z] = ugj$, $e_1 = ug(i)$ and $e_2 = ug(j)$ and the proof proceeds along the same line as the one for Lem. 42. ∎

Lemmas 42 and 43 tell us that, by labelling the prime intervals of $H_D \cong G_{f_0}^\omega$ according to the function l_Δ, one does obtain a (possibly infinite) trace automaton over the finite alphabet $(E, \|)$. The construction of this labelling relies chiefly on Hrushovski's theorem. Nevertheless, we have not yet completely exploited the finite group of automorphisms of $(E, \|)$ generated by $\Psi_1 \ldots \Psi_n$. Let Ω denote that finite group. We now intend to construct from Ω and the representative part of H_D a finite trace automaton $\mathcal{A}$ (over $(E, \|)$) that unfolds to the labelled version of H_D.

We define $\mathcal{A} = (E, \|, Q, q_0, T)$ with set of states Q, initial state q_0, and set of transitions T as follows. Let $Q = \Omega \times V_\Delta$ where $V_\Delta = \{(u, i) \in V_{f_0}^\omega / \ u \in \mathcal{T}_\Delta^\circ\}$, and let $q_0 = <\Psi_\epsilon, (f_0, 1)>$ where Ψ_ϵ is the identity on E. More generally, for $m \in \{1, \ldots n\}^*$, let Ψ_m be inductively defined by $\Psi_{j \cdot m} = \Psi_j \circ \Psi_m$. Then $T \subseteq Q \times E \times Q$ is the set of transitions $\sigma_\Delta(u, i) \xrightarrow{\eta_\Delta(u(k))} \sigma_\Delta(v, j)$ such that $u \in \mathcal{T}_{f_0}$ and uk :

$(u,i) \prec (v,j)$ in H_D, where $\sigma_\Delta : V^\omega_{f_0} \to Q$ is the mapping inductively defined by :

$$\sigma_\Delta(u,j) = \begin{cases} <\Psi_\epsilon,(u,j)> & \textbf{if} \quad u \in \mathcal{T}^\circ_\Delta \\ <\Psi_k \circ \Psi_m,(v,j)> & \textbf{if} \quad u \stackrel{k}{\to} u' \text{ and } \sigma_\Delta(u',j) = <\Psi_m,(v,j)>. \end{cases}$$

We state below a second series of lemmas, necessary for proving that $\mathcal{A}$ unfolds to H_D.

Lemma 44 *Let $(u,i) \in V^\omega_{f_0}$ and $uk : (u,i) \prec (v,j)$ in H_D. If $\sigma_\Delta(u,i) = <\Psi_m,(u',i)>$ then $\eta_\Delta(u(k)) = \Psi_m(u'(k))$.*

Proof: The proof proceeds by induction on the length of the normalisation process for u.

1. **Base Case :** $u \in \mathcal{T}^\circ_\Delta$
   Then $\sigma_\Delta(u,i) = <\Psi_\epsilon,(u,i)>$ and $\eta_\Delta(u(k)) = u(k) = \Psi_\epsilon(u(k))$.

2. **General case :** $u \notin \mathcal{T}^\circ_\Delta$
   Then $u \stackrel{l}{\to} u''$ and if we let $\sigma_\Delta(u'',i) = <\Psi_{m'},(u',i)>$, we obtain from the definition $\sigma_\Delta(u,i) = <\Psi_l \circ \Psi_{m'},(u',i)>$. Now, $\eta_\Delta(u(k)) = \Psi_l(\eta_\Delta(u''(k)) =$ [by inductive assumption] $\Psi_l(\Psi_{m'}(u'(k)) = \Psi_{l \cdot m'}(u'(k))$.

∎

Lemma 45 *For any vertex (u,i) in $V^\omega_{f_0}$, let $E_{(u,i)} = \{\eta_\Delta(u(k)) / \exists v \exists j \cdot uk : (u,i) \prec (v,j)\}$. If $\sigma_\Delta(u_1,i_1) = \sigma_\Delta(u_2,i_2)$ then $E_{(u_1,i_1)} = E_{(u_2,i_2)}$.*

Proof: Let $\sigma_\Delta(u_1,i_1) = <\Psi_{m_1},(u'_1,i_1)>$ and $\sigma_\Delta(u_2,i_2) = <\Psi_{m_2},(u'_2,i_2)>$. Then u_1 and u_2 have the same normal form $u'_1 = u'_2 = u'$, and their respective normalisation processes $u_1 \stackrel{m_1}{\to} u'$ and $u_2 \stackrel{m_2}{\to} u'$ are equivalent in the sense that $\Psi_{m_1} = \Psi_{m_2}$. By Lem. 44, $E_{(u_1,i_1)} = E_{(u_2,i_2)}$. ∎

Corollary 46 *$E_{(u,i)} = \{e \in E | \sigma_\Delta(u,i) \stackrel{e}{\to}\}$, thus the transition relation T depends only upon the pairs (u,i) in V_Δ, and therefore, the definition of $\mathcal{A}$ is constructive.*

Lemma 47 *Let $uk : (u,i) \prec (v,j)$ in H_D. If $u \stackrel{m}{\rightarrow} u'$ for some word $m \in \{1,\ldots,n\}^*$ then $v \stackrel{m}{\rightarrow} v'$ for some v' such that $u'k : (u',i) \prec (v',j)$ in H_D.*

Proof: If $m = \epsilon$, there is nothing to prove. If $m = l \cdot m'$, then $u \stackrel{l}{\rightarrow} u_1 (\stackrel{m'}{\rightarrow} u')$ and we have $u = \beta_l wf$ and $u_1 = \alpha_l wf$. Since $(u,i) \prec (v,j)$, $v = \beta_l wfg$ for some g such that $fi \rightarrow fgj$. Let $v_1 = \alpha_l wfg$, then $v \stackrel{l}{\rightarrow} v_1$ and $(u_1,i) \prec (v_1,j)$. By induction, we can find v' such that $(u',i) \prec (v',j)$ and $v_1 \stackrel{m'}{\rightarrow} v'$ hence $v \stackrel{m}{\rightarrow} v'$. ∎

Proposition 48 *$\mathcal{A} = (E, \|, Q, q_0, T)$ is a finite trace automaton ; and the pair of mappings $(\sigma_\Delta, \eta_\Delta) : H_D \rightarrow \mathcal{A}$ is a folding morphism, where H_D is viewed as an unfolded trace automaton.*

Proof: We verify first that the pair $(\sigma_\Delta, \eta_\Delta)$ satisfies the conditions required from folding morphisms.

1. the initial states are in correspondence: $\sigma_\Delta(f_0,1) = <\Psi_\epsilon, (f_0,1)> = q_0$, by definition of q_0.

2. $uk : (u,i) \prec (v,j)$ in H_D implies $\sigma_\Delta(u,i) \stackrel{\eta_\Delta(u(k))}{\longrightarrow} \sigma_\Delta(v,j)$ by definition of T.

3. For every $(u,i) \in V^\omega_{f_0}$, $\{e \in E | \sigma_\Delta(u,i) \stackrel{e}{\rightarrow}\} =$ [by Lem. 45] $\{\eta_\Delta(u(k)) \mid \exists v \exists j \cdot\ uk : (u,i) \prec (v,j)\} =$ [by Lem. 44] $\{\Psi_m(u'(k)) \mid \exists v \exists j \cdot\ uk : (u,i) \prec (v,j)\}$ where $u \stackrel{m}{\rightarrow} u' \in \mathcal{T}^\circ_\Delta$. Since H_D satisfies axiom (R), which tells that two co-initial prime intervals do not belong to the same event, and since u and u' have the same type, $u(k) \neq u(k')$ and $u'(k) \neq u'(k')$ for $k \neq k'$. Now, Ψ_m is a bijection, and therefore, η_Δ restricts to a bijection between $\{e \in E | \sigma_\Delta(u,i) \stackrel{e}{\rightarrow}\}$ and $\{e \in E_D | \exists k \cdot\ e = u(k) \wedge \exists v \exists j \cdot\ uk : (u,i) \prec (v,j)\}$.

4. every pair of events $e_1, e_2 \in E_D$ which are co-initial at some vertex in H_D (i.e. such that $\exists (u,i)$, (v_1,j_1), $(v_2,j_2) \in V^\omega_{f_0}$ with $[(u,i),(v_1,j_1)] \in e_1$ and $[(u,i),(v_2,j_2)] \in e_2$) are either independent of in conflict hence $e_1 \|_D e_2$ if and only if $\eta_\Delta(e_1) \| \eta_\Delta(e_2)$ by Lem. 42 and Lem. 43.

It remains to prove that $\mathcal{A}$ is indeed a trace automaton.

1. $\mathcal{A}$ is deterministic.

   Let $\sigma_\Delta(u_1, i_1) \stackrel{\eta_\Delta(u_1(k_1))}{\longrightarrow} \sigma_\Delta(v_1, j_1)$ and $\sigma_\Delta(u_2, i_2) \stackrel{\eta_\Delta(u_2(k_2))}{\longrightarrow} \sigma_\Delta(v_2, j_2)$ be two transitions of $\mathcal{A}$, stemming from prime intervals $u_1k_1 : (u_1, i_1) \multimap (v_1, j_1)$ and $u_2k_2 : (u_2, i_2) \multimap (v_2, j_2)$ in H_D, and assume $\sigma_\Delta(u_1, i_1) = \sigma_\Delta(u_2, i_2)$ and $\eta_\Delta(u_1(k_1)) = \eta_\Delta(u_2(k_2))$. Then $i_1 = i_2 =$ [let] i, u_1 and u_2 have the same normal form $u'_1 = u'_2 =$ [let] u' and their normalisation processes $u_1 \stackrel{m_1}{\to} u'$ and $u_2 \stackrel{m_2}{\to} u'$ are equivalent in the sense that $\Psi_{m_1} = \Psi_{m_2} =$ [let] Ψ. By Lem. 47, one can find v'_1 such that $u'k_1 : (u', i) \multimap (v'_1, j_1)$ and $v_1 \stackrel{m_1}{\to} v'_1$, and similarly, one can find v'_2 such that $u'k_2 : (u', i) \multimap (v'_2, j_2)$ and $v_2 \stackrel{m_2}{\to} v'_2$. By definition, $\sigma_\Delta(v_1, j_1) = <\Psi, (v'_1, j_1)>$, and $\sigma_\Delta(v_2, j_2) = <\Psi, (v'_2, j_2)>$. Now, by Lem. 44, $\eta_\Delta(u_1(k_1)) = \Psi(u'(k_1))$ and $\eta_\Delta(u_2(k_2)) = \Psi(u'(k_2))$ thus $u'(k_1) = u'(k_2)$ because Ψ is a bijection. By axiom (R), $(v'_1, j_1) = (v'_2, j_2)$. Therefore $\sigma_\Delta(v_1, j_1) = \sigma_\Delta(v_2, j_2)$.

2. $\mathcal{A}$ satisfies the commutativity condition.
   By Lem. 45 and its corollary, it is sufficient to consider the case of two independent events $e_1 \| e_2$ enabled in state $q = \sigma_\Delta(u, i)$ stemming from prime intervals $u_1k_1 : (u, i) \multimap (v_1, j_1)$ and $u_2k_2 : (u, i) \multimap (v_2, j_2)$ (whence $e_1 = \eta_\Delta(u_1(k_1))$ and $e_2 = \eta_\Delta(u_2(k_2))$). Since $\eta_\Delta(u_1(k_1)) \| \eta_\Delta(u_2(k_2))$, $u_1(k_1) \|_D u_2(k_2)$ by Lem. 43, and therefore there must exist a vertex $(v, j) \in V^\omega_{f_0}$ and two prime intervals $v_1k'_1 : (v_1, j_1) \multimap (v, j)$ and $v_2k'_2 : (v_2, j_2) \multimap (v, j)$ such that $u_1(k_1) = v_2(k'_2)$ and $u_2(k_2) = v_1(k'_1)$. Hence $\sigma_\Delta(v_1, j_1) \stackrel{e_2}{\to} \sigma_\Delta(v, j)$ and $\sigma_\Delta(v_2, j_2) \stackrel{e_1}{\to} \sigma_\Delta(v, j)$ are two transitions in T

∎

This completes the proof for Theo. 19.

5 Conclusion

We have shown in this paper that every context-free event domain is recognizable, i.e. coincides with the domain of configurations of some finite trace automaton. This representation result has been established by a constructive proof, which extracts a finite trace automaton from any graph grammar defining the Hasse diagram of a conflict event domain. The correspondence between "context-free" event structures (i.e. event structures generating context-

free event domains) and finite trace automata may hopefully serve to establish a tight relationship between logics of concurrency in the spirit of [MT92] or [Pen88], and Büchi's monadic second order logic S1S [Büc60]. A related question is to decide whether an event structure presented by rational sets and relations is context-free. Let us stress now some limitations of our work. The finite trace automata which generate context-free event domains form clearly a recursively enumerable and strict subset of the finite trace automata, but we do not know presently whether this set is recursive. For this reason, our representation result fails to establish a full correspondence between well identified classes of models. In another respect, we have failed to characterize the class of event domains which are recognized by finite trace automata. We have set the conjecture that such domains are recognizable if and only if they have finitely many non isomorphic residuals and bounded cliques for relation $\natural$. A first step before proving or disproving this conjecture is to examine the particular case of the (possibly not context-free) distributive event domains, which are known to coincide with the domains of configurations of prime event structures. In a wider prospect, the present work suggests the study of *recognizable graphs* (or at least acyclic graphs) defined in terms of finitely presented groups of automorphisms.

6 Appendices

6.1 A Characterization of Context-free Event Domains

The purpose of this appendix it to give a characterization of context-free event domains. We recall (see Def. 16) that an elementary *zig-zag* for the covering relation $\prec$ between elements of H_D is any sequence $(x_i)_{i\in I}$ of *distinct* elements, indexed by an interval $I \subset \mathbb{N}$ of integers, and such that $x_{2n} \prec x_{2n-1}$ and $x_{2n} \prec x_{2n+1}$ whenever the corresponding indices are in I.

Proposition 49 *A recognizable event domain D is context-free if and only if H_D has bounded elementary zig-zags for the covering relation between its elements.*

Boundedness of elementary zig-zags in the Hasse diagram is clearly a necessary condition. To prove that it is also a sufficient condition we proceed by a series of lemmas.

Lemma 50 *If two finite elements x and y of an event domain D are connected by a zig-zag of the form $x \prec z \prec^{-1} y$ then they are also connected by a path $[z_0, z_1, \ldots, z_{2l}]$ where $x = z_0$, $\forall k \in \{0, \ldots, l-1\}$ $z_k \prec z_{k+1} \prec^{-1} z_{k+2}$ and $z_{2l} = y$.*

Proof : Let $u = e_1, \ldots, e_k$ and $v = e''_1, \ldots, e''_k$ be two sequences of events labelling respective paths from $\perp$ (the minimal element) to x and from $\perp$ to y in T_D, with $[x, z] \in e_k$ and $[y, y] \in e''_k$. Since u and v define the same configuration in $D(T_D)$, they are equivalent by permutations of independent events. Let $u = w_0 \sim w_1 \ldots \sim w_{m-1} \sim w_m = v$ be a proof for $u \sim v$, where w_i, w_{i+1} differ by exactly one permutation and $w_i \in \mathcal{L}(T_D)$ for all i. Set $w_i = w'_i e'_i$ with $e'_i \in E_D$, thus $e'_0 = e_k$ and $e'_m = e''_k$. Denote l the maximal number for which $f(i)$ is defined by $f(0) = 0$ and $f(i+1) = \mathbf{inf}\{j / e'_j \neq e'_{f(i)}\}$. Then $w'_{f(i)} \sim w'_{f(i+1)-1}$ by right cancellation in the commutative monoid. Then for all $i < l$, $w'_{f(i+1)-1} e'_{f(i)} \sim w'_{f(i+1)}$ by permutation of $e'_{f(i)}$ and $e'_{f(i+1)}$ and there must exist $w''_{f(i)}$ such that $w'_{f(i+1)-1} = w''_{f(i)} e'_{f(i+1)}$ and $w'_{f(i+1)} = w''_{f(i)} e'_{f(i)}$. Then we obtain the desired path by setting $\perp \overset{w'_{f(i)}}{\rightarrow} z_{2i}$ and $\perp \overset{w''_{f(i)}}{\rightarrow} z_{2i+1}$ in T_D (for $i < l$). ∎

Lemma 51 *Let x and y be two distinct elements of H_D belonging to the same section S. Then they are connected by a zig-zag $[z_0, z_1, \ldots, z_{2n}]$ i.e. $x = z_0$, $\forall k \in \{0, \ldots, n-1\}$ $z_k \prec z_{k+1} \prec^{-1} z_{k+2}$ and $z_{2n} = y$.*

Proof : By definition of a section, there exists a path $x = x_0 - x_1 - x_2 \ldots - x_m = y$ where $- = (\prec \cup \prec^{-1})$ and x_i lies in the connected component of H_D based on S. If $x_i \in S$ for some $0 < i < m$, then the above path may be split into two strictly shorter paths and the conclusion follows by induction on their length. If $x_i \notin S$ for all $0 < i < m$, then $x_1 - x_2 \ldots - x_{m-1}$ is a path of strictly smaller amplitude (defined as the maximal variation of the distance from the root) in the connected component based on the section which contains x_1 and x_{m-1}, and therefore one may assume by induction on the amplitude of paths the existence of a zig-zag $[y_0, y_1, \ldots, y_{2p}]$ where $x_1 = y_0$, $\forall k \in \{0, \ldots, p-1\}$ $y_k \prec y_{k+1} \prec^{-1} y_{k+2}$ and $y_{2p} = x_{m-1}$. Now the conclusion follows by Lem. 50,

applied to each $y_{2i} \prec y_{2i+1} \prec^{-1} y_{2i+2}$. ∎

We may assume without loss of generality that the elements z_i are all distinct, i.e.

Corollary 52 *Let x and y be two distinct elements of H_D belonging to the same section S. Then they are connected by an elementary zig-zag $[z_0, z_1, \ldots, z_{2n}]$ i.e. $x = z_0$, $\forall k \in \{0,\ldots,n-1\}$ $z_k \prec z_{k+1} \prec^{-1} z_{k+2}$ and $z_{2n} = y$.*

Lemma 53 *Let $\mathcal{A}$ be a finite trace automaton such that all elementary zig-zags $z_0 \prec z_1 \prec^{-1} z_2 \ldots \prec z_{2n-1} \prec^{-1} z_{2n}$ in $H_{D(\mathcal{A})}$ have bounded length $n < N$, then the size of the sections of $H_{D(\mathcal{A})}$ is bounded by $K = (p(q-1))^{N+1}$ where p is the out-degree of $\mathcal{A}$ and q is the size of the alphabet of $\mathcal{A}$.*

Proof: For a given x , the number of y 's distinct from x and such that $x \prec z \prec^{-1} y$ is bounded by $p(q-1)$. Thus K is an upper bound to the number of elementary zig-zags $z_0 \prec z_1 \prec^{-1} z_2 \ldots$ from a fixed origin z_0. The conclusion follows by Cor. 52. ∎

Lemma 54 *Let $\mathcal{A}$ be a finite trace automaton such that $H_{D(\mathcal{A})}$ has bounded sections, then $H_{D(\mathcal{A})}$ is a context-free graph.*

Proof: Let $D = D(\mathcal{A})$. For any element x of an event domain D, we let D_x denote the restriction of D to the elements greater than x . Similarly, for any section S of D, we let D_S denote the connected component of H_D based on S, i.e. the restriction of D to the set $\bigcup_{x \in S} D_x$. Let $S = \{x_1, \ldots, x_k\}$ be a section of D then the ordered set D_S is determined up to isomorphism by the coproduct $\coprod_{1 \le i \le k} D_{x_i}$ together with the equivalence relation on the set of elements of *depth* 1 of $\coprod_{1 \le i \le k} D_{x_i}$ consisting of the pairs $(i, y) \equiv (j, y)$ for $y \in D_{x_i} \cap D_{x_j}$.

> Actually, we first notice that D_S is the amalgamated sum of the D_{x_i} defined as follows. We consider the equivalence on the coproduct $\coprod_{1 \le i \le k} D_{x_i}$ consisting of the pairs $(i, y) \equiv (j, y)$ for $y \in D_{x_i} \cap D_{x_j}$. The prefix relation associated with this equivalence is given by $(i, y)(\le \cup \equiv)^*(j, y')$ with arbitrary i and j and $y \le y'$ in D. Therefore $\equiv$ is the equivalence induced by this preorder and $D_S \cong ((\coprod_{1 \le i \le k} D_{x_i})/\equiv, (\le \cup \equiv)^*/\equiv)$.

Now since D is an event domain $(i,y) \equiv (j,y)$ if and only if $x_i \uparrow x_j$ (i.e. $D_{x_i} \cap D_{x_j} \neq \emptyset$) and $x_i \vee x_j \leq y$ moreover $x_i \prec\!\!\!\cdot\, (x_i \vee x_j) \prec\!\!\!\cdot^{-1} x_j$. Hence the equivalence relation $\equiv$ is fully characterized by its restriction to the elements of depth 1.

Therefore if the size of sections is bounded by K, and if we let n and m be the respective number of states and out-degree of $\mathcal{A}$, the number of non isomorphic components of H_{D_S} is bounded by $(n+1)^K (mK)^{\equiv}$ where $l^{\equiv}$ is the number of equivalence relations on the set $\{1,\ldots,l\}$. ∎

6.2 Deciding whether a Graph Grammar Generates a Conflict Event Domain

This appendix is devoted to the proof of the following theorem.

Theorem 55 *Given a monotone and uniform graph grammar G and a unary hyperarc symbol f_0, one can decide whether the generated context-free graph $G^{\omega}_{f_0}$ is the transitive reduction of the order on finite elements in some conflict event domain.*

We state in section 6.2.1 a characterization of those graphs that are isomorphic to transitive reductions of conflict event domains. We show in section 6.2.2 that this sub-class of graphs is finitely axiomatized by monadic second order formulas within the class of context-free graphs. Theo. 55 then follows from a theorem of Müller and Schupp, stating the decidability of the monadic theory of every context-free graph. We give in section 6.2.3 a more direct proof of Theo. 55, that relies solely on the computation of fixed points of monotone operators on finite lattices and leads therefore to practical algorithms with lower complexity.

6.2.1 Transitive Reductions of Conflict Event Domains

We recall that an event domain is a finitary Scott domain in which the finite elements satisfy the three axioms (C, R, and V). In fact, any Scott domain (i.e. ω-algebraic and bounded complete CPO) is completely determined by its restriction to finite elements. In the sequel, the set of finite elements of a Scott

domain equipped with the induced order is called the *compact core* of that domain.

Proposition 56 *A partial order is the compact core of a Scott domain if and only if it is countable, has a least element and is bounded complete.*

For the sake of completeness we sketch the proof of this proposition. The reader will easily fill in the gaps but may also consult [Plo83] for more details.

Let $\mathbf{F}_{in}(D)$ denote the compact core of an ω-algebraic CPO D. The *completion by ideals* of an ordered set X, denoted $\mathbf{I}(X)$, is the set of ideals of X ordered by inclusion. An *ideal* of X is a *non-empty* subset $I \subset X$ which is *directed* ($\forall x, y \in I \ \exists z \in I \ . \ x \leq z \geq y$) and *downward closed* ($\forall x \in I \ \forall y \in X \ y \leq x \Rightarrow y \in I$).

Lemma 57 *Let F be a countable partial order with a least element, then $D = \mathbf{I}(F)$ is an ω-algebraic CPO whose compact core is isomorphic to F. The isomorphism $F \cong \mathbf{F}_{in}(D)$ maps any element $x \in F$ to the principal ideal $\downarrow x = \{y \in D / y \leq x\}$. Conversely, let D be an ω-algebraic CPO, then $F = \mathbf{F}_{in}(D)$ is a countable partial order with a least element and its completion by ideals is isomorphic to D. The isomorphism $D \cong \mathbf{I}(F)$ maps any element $x \in D$ to the ideal $\{y \in \mathbf{F}_{in}(D) / y \leq x\}$ and maps any ideal to its least upper bound in D.*

Proof: The completion by ideals of an ordered set F is a *pre-CPO* (i.e. an ordered set in which any *non-empty* directed subset has a least upper bound) because the union of any non-empty directed family of ideals is an ideal (hence it is the least upper bound of that family). For the same reason, any principal ideal $\downarrow x = \{y \in D / y \leq x\}$ is finite. If F has a least element $\perp$, then its completion by ideals is a CPO whose least element is the principal ideal generated by $\perp$, let $\downarrow \perp = \{\perp\}$. If F is moreover countable, then any ideal I of F admits a co-final increasing ω-chain, i.e. there exists a sequence $(x_n)_{n \in \omega}$ such that $\forall n. \ x_n \leq x_{n+1}$ and $I = \bigcup_{n \in \omega} \downarrow x_n$, whence any finite element in $\mathbf{I}(F)$ is a principal ideal. Thus $D = \mathbf{I}(F)$ is ω-algebraic. The second part of the lemma is easily verified. ∎

Prop. 56 follows from Lem. 57 and the following observation.

Observation 58 *An ω-algebraic CPO is bounded complete if and only if its compact core is bounded complete.*

Corollary 59 *A directed graph $(X, \prec)$ with a countable set of vertices X is the transitive reduction of the compact core of a confict event domain if and only if the reflexive and transitive closure $\leq = \prec^*$ of the incidence relation*

(i)	*is an order*	*i.e.*	*the graph is acyclic*
(ii)	*with a least element*	*i.e.*	*the graph is rooted*
(iii)	*which is finitary*	*i.e.*	$\forall x \downarrow x = \{y / y \leq x\}$ *is finite*

and the folowing three axioms are satisfied:

Axiom C: $(x \prec y,\ x \prec z,\ y \neq z,\ y \uparrow z) \Rightarrow (y \vee z \text{ exists},\ y \prec (y \vee z),\ z \prec (y \vee z))$
Axiom R: $[x, y] \succ\!\prec [x, y'] \Rightarrow y = y'$
Axiom V: $[x, x'] \succ\!\prec [y, y'] \ \&\ [x, x''] \succ\!\prec [y, y''] \ \&\ x' \uparrow x'' \Rightarrow y' \uparrow y''$

Corollary 60 *Let G be a monotone and uniform graph grammar and let f_0 be a unary hyperarc symbol, then the graph $G^\omega_{f_0}$ is isomorphic to the transitive reduction of the compact core of a conflict event domain if and only if it satisfies axioms the three axioms (C), (R) and (V).*

6.2.2 Monadic Second Order Logic and Conflict Event Domains

All the conditions stated in Cor. 59 with the exception of condition *(iii)* may be expressed by monadic second order formulas over a signature $\{\prec\}$ with a single predicate symbol of arity two. The formulas in **MSOL**$(\prec)$ are constructed from a countable set of variables x, y, $z \ldots$ for individuals and from a countable set of variables X, Y, $Z \ldots$ for sets of individuals (i.e. monadic predicates) by combining atomic formulas $x \in X$ and $x \prec y$ with the boolean connectives and the first order and second order quantifiers. A *model* for **MSOL**$(\prec)$ is a structure $\mathbf{G} = (V, \prec)$ where $\prec$ is a binary relation over the universe V, hence $\mathbf{G}$ is a directed graph. The relation of validation $\mathbf{G} \models \varphi$ between models and formulas follows the usual definition of validation for second order formulas, i.e. variables $x, y, z \ldots$ are assigned to vertices, variables $X, Y, Z \ldots$ are assigned to sets of vertices, and the predicate symbol $\prec$ is interpreted by the incidence relation of the graph. The monadic theory of a graph $\mathbf{G} = (V, \prec)$ is the set of formulas of **MSOL**$(\prec)$ which are satisfied in that model.

Theorem 61 (Müller and Schupp) *The monadic theory of a context-free graph is decidable.*

This theorem means that given any recursive presentation of a context-free graph $\mathbf{G}$, for instance a grammar (G, M) such that $\mathbf{G} = G^{\omega}(M)$, and a formula $\varphi \in \mathbf{MSOL}(\prec)$, one can decide whether $\mathbf{G} \models \varphi$. Therefore a proof for Theo. 55 can be derived from the following proposition.

Proposition 62 *The sub-class of the context-free graphs which are isomorphic to transitive reductions of compact cores of conflict event domains has a finite axiomatization in* $\mathbf{MSOL}(\prec)$.

The rest of this subsection is dedicated to the proof of Prop. 62. The acyclic context-free graphs satisfy naturally the condition *(iii)* stated in Cor. 59. It only remains to find the formulas in $\mathbf{MSOL}(\prec)$ which express the other conditions. Observe that the reflexive and transitive closure of a binary relation which may be defined in $\mathbf{MSOL}(\prec)$ is also definable in $\mathbf{MSOL}(\prec)$:

$$xR^*y \quad \equiv \quad \forall X[(\forall z \forall u(z \in X \wedge zRu \Rightarrow u \in X)) \wedge x \in X] \Rightarrow y \in X$$

Therefore the relation $\leq = \prec^*$ is definable in $\mathbf{MSOL}(\prec)$, as well as the following relations :

$$\begin{array}{lcl}
x < y & \equiv & x \leq y \ \ \& \ \ x \neq y \\
x \prec y & \equiv & x < y \ \ \& \ \ (x \leq z \leq y \Rightarrow (z = x \ \text{ or } \ z = y)) \\
x \uparrow y & \equiv & \exists z \ \ x \leq z \ \ \& \ \ y \leq z \\
x = y \vee z & \equiv & y \leq x \ \ \& \ \ z \leq x \ \ \& \ \ [\forall u \ \ (y \leq u \ \ \& \ \ z \leq u) \Rightarrow x \leq u] \\
x = y \wedge z & \equiv & x \leq y \ \ \& \ \ x \leq z \ \ \& \ \ [\forall u \ \ (u \leq y \ \ \& \ \ u \leq z) \Rightarrow u \leq x] \\
[x, y] \prec [z, u] & \equiv & x \prec y \ \ \& \ \ z \prec u \ \ \& \ \ x \prec z \ \ \& \ \ y \prec u \\
[x, y] \succ\!\prec^1 [z, u] & \equiv & [x, y] \prec [z, u] \ \text{ or } \ [z, u] \prec [x, y]
\end{array}$$

Unfortunately, the relation of projectivity $\succ\!\prec = (\succ\!\prec^1)^*$ has no straightforward definition in $\mathbf{MSOL}(\prec)$: since $\succ\!\prec^1$ is a relation between *pairs* of individuals, quantifiers over *binary* predicate variables $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \ldots$ are introduced when expressing $[x, y] \succ\!\prec [z, u]$ as

$$\forall \mathcal{X}[(\forall x_1 \forall x_2 \forall y_1 \forall y_2 (x_1 \mathcal{X} y_1 \wedge [x_1, y_1] \succ\!\prec^1 [x_2, y_2] \Rightarrow x_2 \mathcal{X} y_2)) \wedge x \mathcal{X} y] \Rightarrow z \mathcal{X} u$$

Nevertheless, each bounded relation $\succ\!\prec^{\leq n} = \bigcup_{k\leq n} \succ\!\prec^k$ may be defined in **MSOL**$(\prec)$, leading to a *countable* axiomatization of the conditions imposed by axioms R and V on conflict event domains:

Axiom R_n: $\quad [x,y] \succ\!\prec^{\leq n} [x,y'] \Rightarrow y = y'$
Axiom $V_{n,m}$: $\quad [x,x'] \succ\!\prec^{\leq n} [y,y'] \;\&\; [x,x''] \succ\!\prec^{\leq m} [y,y''] \;\&\; x' \uparrow x'' \Rightarrow y' \uparrow y''$

Now, the order on prime intervals $\leq \;= \prec^*$ is definable in **MSOL**$(\prec)$, since it may be characterized as:

$$[x,y] \leq [z,u] \Leftrightarrow (x \prec y \;\;\&\;\; z \prec u \;\;\&\;\; x = y \wedge z \;\;\&\;\; u = y \vee z)$$

The above axioms R_n and $V_{n,m}$ may therefore be replaced by the following:

Axiom R'_n: $\quad [x,y]\ (\leq \cup \geq)^n [x,y'] \Rightarrow y = y'$
Axiom $V'_{n,m}$: $\quad [x,x']\ (\leq \cup \geq)^n [y,y'] \;\&\; [x,x'']\ (\leq \cup \geq)^m [y,y''] \;\&\; x' \uparrow x'' \Rightarrow y' \uparrow y''$

At this point, Theo. 55 would follow by direct application of Theo. 61 from a positive answer to the following question:

Question 63 *Given a uniform graph grammar generating an acyclic graph* $\mathbf{G} = (V, \prec)$*, can we find an integer* N *such that* $\succ\!\prec = (\leq \cup \geq)^N$ *?*

Unfortunately, Figure. 14 displays a context-free event domain where unbounded zig-zags must be followed for connecting pairs of projective prime intervals, showing that such an N cannot exist in general.
The three lemmas which are established below show that the conditions imposed by axioms (R) and (V) on conflict event domains may however be expressed by a *finite* set of **MSOL**$(\prec)$ formulas.

Lemma 64 *There exists a formula* **finite**(X) *of* **MSOL**$(\prec)$ *which is satisfied in a model* $G{=}(V,\prec)$ *given by an acyclic graph of finite degree if and only if* X *is a finite subset of* V*.*

Proof: The following formulas express respectively that the set X admits an infinite chain, and that the set X is downward closed :

Infinite-Path$(X) \quad \equiv \quad \exists x\ (x \in X) \;\wedge\; [\forall x\ (x \in X) \Rightarrow (\exists y \;\cdot\; y \in X \;\;\wedge\;\; x \prec y]$
Downward-Closed$(X) \quad \equiv \quad \forall x\ \forall y\ \ [x \prec y \;\;\wedge\;\; y \in X] \Rightarrow x \in X$

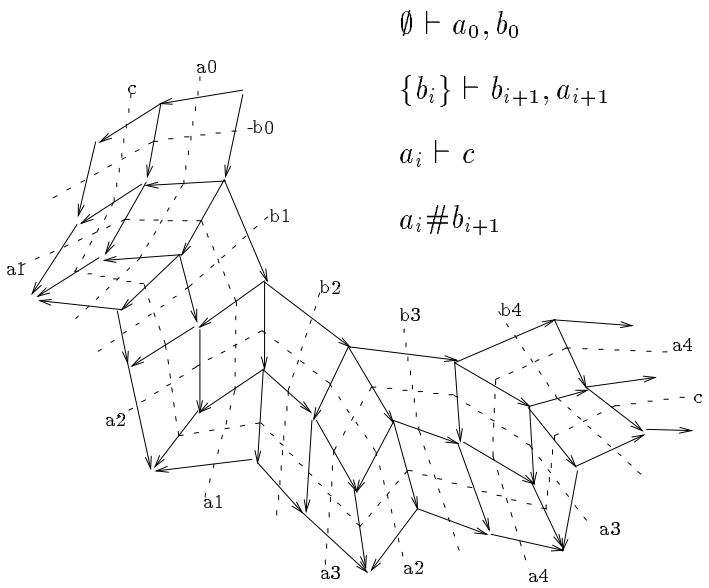


Figure 14: a context-free event domain with unbounded optimal zig-zags between prime intervals

In view of Koenig's lemma, the following formula fulfils the requirement of the lemma.

$$\textbf{finite}(X) \;\equiv\; \exists Y\ X \subseteq Y \;\wedge\; \textbf{Downward-Closed}(Y) \;\wedge\; \neg(\textbf{Infinite-Path}(Y))$$

∎

Restricting ourselves to models of $\mathbf{MSOL}(\prec)$ given by rooted and directed acyclic graphs of finite degree, let us introduce the $\mathbf{MSOL}(\prec)$ formula $X \prec Y$ defined by:

$$\begin{array}{rcl} X \prec Y & \equiv & \textbf{finite}(X) \;\wedge\; \textbf{finite}(Y) \\ & & \wedge\quad (\forall x \in X\ \ \exists!\, y \in Y\ \ x \prec y) \wedge\ \ (\forall y \in Y\ \ \exists!\, x \in X\ \ x \prec y) \\ & & \wedge\quad (\forall X' \subset X\ \ \exists x' \in X'\, \exists x \in X \setminus X'\, \exists y \in Y\, \exists y' \in Y \\ & & \qquad [x,y] \succ\!\prec^1 [x',y']) \end{array}$$

then the predicate $\textbf{Proj}(x,y,x',y')$ defined by:

$$\textbf{Proj}(x,y,x',y') \;\equiv\; \exists X\, \exists Y\ \ X \prec Y \;\wedge\; \left\{ \begin{array}{l} x \in X \;\wedge\; x' \in X \;\wedge\; y \in Y \wedge\; y' \in Y \\ \wedge\, x \prec y \;\wedge\; x' \prec y' \end{array} \right\}$$

entails when it is satisfied that the prime intervals $[x,y]$ and $[x',y']$ are projective ($[x,y] \succ\!\prec [x',y']$). The following lemma establishes a weakening of the converse implication.

Lemma 65 *Let G be a rooted directed acyclic graph of finite degree. If G satisfies the following condition:*

$$\textbf{Strong-R} \;\equiv\; \forall X\, \forall Y\ \ X \prec Y \;\Rightarrow\; \left\{ \begin{array}{l} \forall x\, \forall y\, \forall x'\, \forall y' \\ (x \in X \;\wedge\; y \in Y \;\wedge\; [x,y] \succ\!\prec^1 [x',y']) \\ \Rightarrow \left[\begin{array}{l} [x' \notin X \Rightarrow \forall y'' \in Y \;\cdot\; \textbf{not}(x' \prec y'')] \\ \wedge\, [y' \notin Y \Rightarrow \forall x'' \in X \;\cdot\; \textbf{not}(x'' \prec y')] \end{array} \right] \end{array} \right\}$$

then the formula $\textbf{Proj}(x,y,x',y')$ holds in G if and only if $[x,y] \succ\!\prec [x',y']$.

Proof: Assume $[x,y] \succ\!\prec [x',y']$ and consider a chain of projective prime intervals $[x,y] = [x_0,y_0] \succ\!\prec^1 [x_1,y_1] \ldots \succ\!\prec^1 [x_n,y_n] = [x',y']$. Then by condition **Strong-R**, all the sets $X_i = \{x_0,\ldots,x_i\}$ and $Y_i = \{y_0,\ldots,y_i\}$ for $0 \leq i \leq n$ are such that $X_i \prec Y_i$. The formula $\textbf{Proj}(x,y,x',y')$ is then satisfied with $X = X_n$

and $Y = Y_n$. ∎

Therefore if G satisfies condition **Strong-R**, the relation of projectivity between prime intervals may be expressed by a monadic second order formula ; and in that case axioms (R) and (V) may also be expressed by **MSOL**($\prec$) formulas. Note by the way that if **Strong-R** holds in G then necessarily axiom (R) is satisfied by G. The only thing which remains to be proved is that this condition **Strong-R** is not too demanding, namely:

Lemma 66 *The compact core of a conflict event domain satisfies condition* **Strong-R**.

Proof: $X \prec Y$ implies that X and Y are finite sets of configurations such that $Y = \{x \cup \{e\} \mid x \in X\}$ where e is an event that does not occur in any configuration of X: $\forall x \in X \ e \notin x$. Thus, symmetrically, we also have $X = \{y \setminus \{e\} \mid y \in Y\}$ and the event e occurs is every configuration of Y. Now suppose that x, y, x', and y' are configurations such that $x \in X$, $y \in Y$, and $[x, y] \succ\!\prec^1 [x', y']$. Thus $y' = x' \cup \{e\}$. Let us assume $\exists y'' \in Y \cdot x' \prec y''$, then necessarily $y'' = x' \cup \{e\}$ because $e \notin x'$ (since $x' = y' \setminus \{e\}$) and $e \in y''$ (since $y'' \in Y$). Then $x' \in X$ since it is a configuration of the form $x' = y'' \setminus \{e\}$ with $y'' \in Y$, (then also $y' \in Y$ since $y' = x' \cup \{e\} = y''$). Symmetrically, $\exists x'' \in X \cdot x'' \prec y'$ implies $y' \in Y$. ∎

This achieve the proof of Prop. 62 and thus of Theo. 55.

6.2.3 An alternative decision for Axioms (C), (R), and (V)

We give in this section a direct proof of Theo. 55, that does not rely on monadic second order logic and leads to efficient algorithms for the decision of axioms (C), (R), and (V). Assume given a uniform and monotone graph grammar G. Thus, if f_0 is a unary hyperarc symbol, the associated graph $G^\omega_{f_0}$ is the transitive reduction of a finitary order with least element. Remind that $u(i)$, for $u \in \mathcal{T}^g_f$ and $1 \leq i \leq n(g)$, denotes the event in G^ω_f which the prime interval $u \cdot i$ belongs to. Remind also that the projective order on prime intervals is the equivalence generated by the suffix rewriting system with finite set of rules $\{g \cdot i \to gh \cdot j / \ g \cdot i \prec gh \cdot j \text{ in } G^\omega_g\}$. This can be rephrased as follows.

Table 1: the rules for projectivity in S_f

$\dfrac{g \cdot i \to gh \cdot j \qquad u \in \mathcal{T}_f^g}{u \cdot i \prec\!\!\succ_f uh \cdot j}$	$\dfrac{u \cdot i \prec\!\!\succ_f v \cdot j}{v \cdot j \prec\!\!\succ_f u \cdot i}$
$\dfrac{u \in \mathcal{T}_f^g \qquad 1 \leq i \leq n(g)}{u \cdot i \prec\!\!\succ_f u \cdot i}$	$\dfrac{u \cdot i \prec\!\!\succ_f v \cdot j \qquad v \cdot j \prec\!\!\succ_f w \cdot k}{u \cdot i \prec\!\!\succ_f w \cdot k}$

Lemma 67 *Two prime intervals $u \cdot i$ and $v \cdot j$ in G_f^ω belong to the same event $u(i)=v(j)$ if and only if the relation $u \cdot i \prec\!\!\succ_f v \cdot j$ is provable from the inference rules given in Table 1.*

Definition 68 (Relations of Local Independence and Local Conflict)

Given a uniform and monotone graph grammar G, and a pair of prime intervals $g \cdot i = [x, y]$ and $g \cdot j = [x, z]$ in G_g^ω, let $g \cdot i \parallel^0 g \cdot j$ if there exists a diamond $\Diamond(x, y, z, t)$ in G_g^ω, otherwise let $g \cdot i \#^0 g \cdot j$.

Since it suffices to check the graph G_g^ω up to depth 2 in order to determine whether $g \cdot i \parallel^0 g \cdot j$ or $g \cdot i \#^0 g \cdot j$, the relations of local independence ($\parallel^0$) and local conflict ($\#^0$) can be easily computed from the graph grammar G. Fig. 15 shows for instance the relations of local independence and conflict for the uniform and monotone graph grammar which has been taken as a running example in section 4.

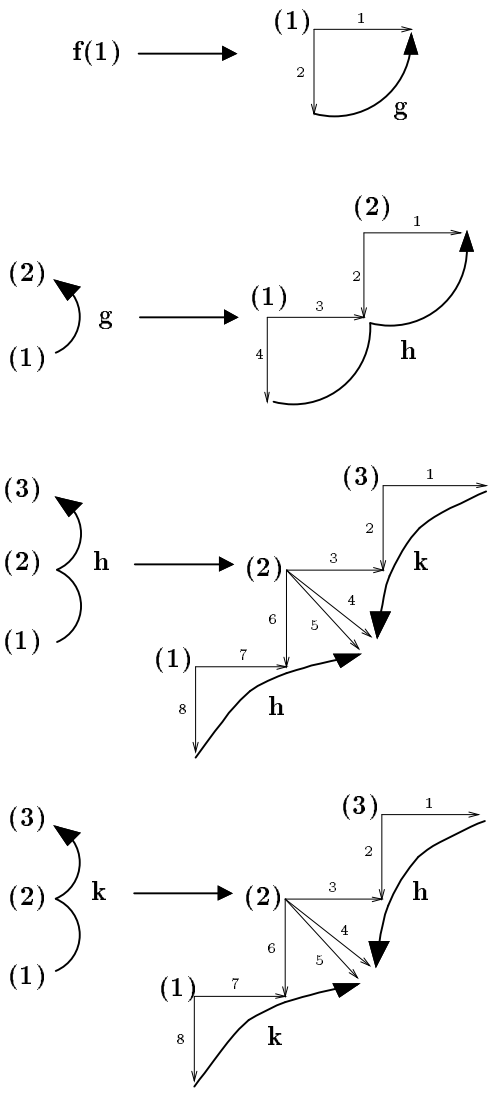
Proposition 69 *Let $u \cdot i$ and $v \cdot j$ be prime intervals in G_f^ω, then:*

$$u(i) = v(j), \qquad \textit{resp. } u(i) \| v(j), \qquad \textit{resp. } u(i) \# v(j)$$

if and only if the relation

$$u \cdot i \prec\!\!\succ_f v \cdot j, \qquad \textit{resp. } u \cdot i \parallel_f v \cdot j, \qquad \textit{resp. } u \cdot i \#_f v \cdot j$$

is provable from the instances of the relations of local independence and conflict (taken as axioms) and the inference rules given in Tables 1 and 2.



$$f \cdot 1 \parallel^0 f \cdot 2$$

$$g \cdot 1 \parallel^0 g \cdot 2 \qquad g \cdot 3 \parallel^0 g \cdot 4$$

$$h \cdot 1 \parallel^0 h \cdot 2 \qquad h \cdot 3 \#^0 h \cdot 5$$
$$h \cdot 3 \parallel^0 h \cdot 4 \qquad h \cdot 3 \#^0 h \cdot 6$$
$$h \cdot 5 \parallel^0 h \cdot 6 \qquad h \cdot 4 \#^0 h \cdot 5$$
$$h \cdot 7 \parallel^0 h \cdot 8 \qquad h \cdot 4 \#^0 h \cdot 6$$

$$k \cdot 1 \parallel^0 k \cdot 2 \qquad k \cdot 3 \#^0 k \cdot 5$$
$$k \cdot 3 \parallel^0 k \cdot 4 \qquad k \cdot 3 \#^0 k \cdot 6$$
$$k \cdot 5 \parallel^0 k \cdot 6 \qquad k \cdot 4 \#^0 k \cdot 5$$
$$k \cdot 7 \parallel^0 k \cdot 8 \qquad k \cdot 4 \#^0 k \cdot 6$$

Figure 15: a uniform and monotone graph grammar and the associated relations of local independence and conflict

Table 2: the rules for independence and conflict in S_f

rules for independence	rules for conflict
$\dfrac{g\cdot i \parallel^0 g\cdot j \qquad u\in \mathcal{T}_f^g}{u\cdot i \parallel_f u\cdot j}$	$\dfrac{g\cdot i \#^0 g\cdot j \qquad u\in \mathcal{T}_f^g}{u\cdot i \#_f u\cdot j}$
$\dfrac{u\cdot i \prec\!\!\succ_f v\cdot j \qquad v\cdot j \parallel_f w\cdot k}{u\cdot i \parallel_f w\cdot k}$	$\dfrac{u\cdot i \prec\!\!\succ_f v\cdot j \qquad v\cdot j \#_f w\cdot k}{u\cdot i \#_f w\cdot k}$
$\dfrac{u\cdot i \parallel_f v\cdot j \qquad v\cdot j \prec\!\!\succ_f w\cdot k}{u\cdot i \parallel_f w\cdot k}$	$\dfrac{u\cdot i \#_f v\cdot j \qquad v\cdot j \prec\!\!\succ_f w\cdot k}{u\cdot i \#_f w\cdot k}$

The above proposition states that the deductive system S_f defined by Tables 1 and 2 is sound and complete for the relations of projectivity, independence and conflict in G_f^ω. However S_f gives rise to an infinite theory and cannot be used to decide on $u(i) = v(j)$, $u(i)\|v(j)$ or $u(i)\#v(j)$. In order to remedy this drawback, let us consider as a possible alternative to S_f the finite deductive system S formed of all instances of the relations of local independence and conflict, taken as axioms, plus the inference rules given in Table. 3.

Proposition 70 *Let $u\cdot i$ and $v\cdot j$ be prime intervals in G_f^ω, then :*

$$f(i) = f(j), \quad \textit{resp. } f(i)\|f(j), \quad \textit{resp. } f(i)\#f(j)$$
$$\textit{if and only if the relation}$$
$$f\cdot i \prec\!\!\succ f\cdot j, \quad \textit{resp. } f\cdot i \,\|f\,\cdot j, \quad \textit{resp. } f\cdot i \,\#\, f\cdot j$$

is provable in S.

Proof:

"if" part: clear, since the set $\{f\cdot i \mid f\in\Sigma \wedge i\le n(f)\}$, equipped with relations $\prec\!\!\succ = \bigcup_{f\in\Sigma} \prec\!\!\succ_f$, $\| = \bigcup_{f\in\Sigma} \|_f$, and $\# = \bigcup_{f\in\Sigma} \#_f$, is a model of S.

"only if" part: Assume $f(i) \parallel f(j)$. By Prop. 69, $f\cdot i \parallel_f f\cdot j$ has a proof in

Table 3: the rules of S

Basic rules

$$\frac{1 \leq i \leq n(f)}{f \cdot i \succ\!\prec f \cdot i} \qquad \frac{f \cdot i \parallel^0 f \cdot j}{f \cdot i \parallel f \cdot j} \qquad \frac{f \cdot i \#^0 f \cdot j}{f \cdot i \# f \cdot j}$$

Propagation rules

$$\frac{g \cdot i \to gh \cdot j \qquad g \cdot i' \to gh \cdot j' \qquad h \cdot j \succ\!\prec h \cdot j'}{g \cdot i \succ\!\prec g \cdot i'}$$

$$\frac{g \cdot i \to gh \cdot j \qquad g \cdot i' \to gh \cdot j' \qquad h \cdot j \parallel h \cdot j'}{g \cdot i \parallel g \cdot i'}$$

$$\frac{g \cdot i \to gh \cdot j \qquad g \cdot i' \to gh \cdot j' \qquad h \cdot j \# h \cdot j'}{g \cdot i \# g \cdot i'}$$

Saturation rules

$$\frac{f \cdot i \succ\!\prec f \cdot j \qquad f \cdot j \succ\!\prec f \cdot k}{f \cdot i \succ\!\prec f \cdot k}$$

$$\frac{f \cdot i \succ\!\prec f \cdot j \qquad f \cdot j \parallel f \cdot k}{f \cdot i \parallel f \cdot k} \qquad \frac{f \cdot i \succ\!\prec f \cdot j \qquad f \cdot j \# f \cdot k}{f \cdot i \# f \cdot k}$$

$$\frac{f \cdot i \parallel f \cdot j \qquad f \cdot j \succ\!\prec f \cdot k}{f \cdot i \parallel f \cdot k} \qquad \frac{f \cdot i \# f \cdot j \qquad f \cdot j \succ\!\prec f \cdot k}{f \cdot i \# f \cdot k}$$

S_f. We show by induction on the length of this proof that $f \cdot i \parallel f \cdot j$ has a proof in S. The cases for $\#$ and $\succ\!\!\prec$ are similar.

1. **Base case:** $f \cdot i \parallel_f f \cdot j$ because $f \cdot i \parallel^0 f \cdot j$.
   Then $f \cdot i \parallel f \cdot j$ is proved in S by the application of a basic rule.

2. **General Case:** $f \cdot i \parallel_f f \cdot j$ because there exist two sequences of prime intervals $(u_k \cdot i_k)_{1 \leq k \leq n}$ and $(v_k \cdot j_k)_{1 \leq k \leq m}$ such that $u_1 = v_1 = f$, $i_1 = i$, $j_1 = j$, $u_n = v_m = w \cdot g$ and $g \cdot i_n \parallel^0 g \cdot j_m$, where $u_k \cdot i_k \prec u_{k+1} \cdot i_{k+1}$ or $u_{k+1} \cdot i_{k+1} \prec u_k \cdot i_k$ for $1 \leq k < n$, and $v_k \cdot i_k \prec v_{k+1} \cdot i_{k+1}$ or $v_{k+1} \cdot i_{k+1} \prec v_k \cdot i_k$ for $1 \leq k < m$. Since u_{k+1} is formed either by adding one letter to u_k or by removing its last letter, the sequence $(u_k \cdot i_k)_{1 \leq k \leq n}$ may be split to p subsequences $(u_k \cdot i_k)_{n_l \leq k < n_{l+1}}$ [with $n = n_{p+1} - 1$] such that $u_{n_l} = f$ for all $l \leq p$ and $u_k = f h_l \cdot w_k$ with $h_l \in \mathbf{succ}(f)$ for $n_l < k < n_{l+1}$. The sequence $(v_k \cdot j_k)_{1 \leq k \leq m}$ may be split in a similar way to q subsequences $(v_k \cdot j_k)_{m_l \leq k < m_{l+1}}$ [with $m = m_{q+1} - 1$]. Two cases can occur.

   (a) $p = q = 1$
   Then for $1 < k \leq n$, u_k is of the form $fh \cdot w_k$ and for $1 < k \leq m$, v_k is of the form $fh' \cdot w'_k$. Since $u_n = v_m = w \cdot g$, $h = h'$. Let us simplify all words on the left by f in the sequences $(u_k \cdot i_k)_{2 \leq k \leq n}$ and $(v_k \cdot j_k)_{2 \leq k \leq m}$, then we obtain an S_h-proof for $h \cdot i_2 \parallel_h h \cdot j_2$, strictly shorter than the S_f-proof for $f \cdot i \parallel_f f \cdot j$. By the induction hypothesis, $h \cdot i_2 \parallel h \cdot j_2$ is provable in S, whence $f \cdot i \parallel f \cdot j$ can be derived by one application of the propagation rule for independence in S.

   (b) $p + q > 2$
   The following hold by induction hypothesis: for every $l < p$, the subsequence $(u_k \cdot i_k)_{n_l \leq k \leq n_{l+1}}$ may be mapped to an S-proof for $u_{n_l} \cdot i_{n_l} \succ\!\!\prec u_{n_{l+1}}$
   $\cdot i_{n_{l+1}}$, for every $l < q$, the subsequence $(v_k \cdot j_k)_{m_l \leq k \leq m_{l+1}}$ may be mapped to an S-proof for $v_{m_l} \cdot j_{m_l} \succ\!\!\prec v_{m_{l+1}} \cdot j_{m_{l+1}}$, and the remaining subsequences $(u_k \cdot i_k)_{n_p \leq k < n_{p+1}}$ and $(v_k \cdot j_k)_{m_q \leq k < m_{q+1}}$ may be mapped jointly to an S-proof for $u_{n_p} \cdot i_{n_p} \parallel v_{m_q} \cdot j_{m_q}$. The relation $f \cdot i \parallel f \cdot j$ may be derived therefrom by repeated applications of the saturation rule for independence in S.

Table 4: the rules for projectivity in S_f^n

$$\frac{g \cdot i \asymp g \cdot j \qquad u \in \mathcal{T}_f^g \qquad |u| = n}{u \cdot i \asymp_f^n u \cdot j}$$

$$\frac{g \cdot i \to gh \cdot j \qquad u \in \mathcal{T}_f^g \qquad |u| < n}{u \cdot i \asymp_f^n uh \cdot j} \qquad \frac{u \cdot i \asymp_f^n v \cdot j}{v \cdot j \asymp_f^n u \cdot i}$$

$$\frac{u \in \mathcal{T}_f^g \qquad |u| < n \qquad 1 \leq i \leq n(g)}{u \cdot i \asymp_f^n u \cdot i} \qquad \frac{u \cdot i \asymp_f^n v \cdot j \qquad v \cdot j \asymp_f^n w \cdot k}{u \cdot i \asymp_f^n w \cdot k}$$

∎

Observe that the set of S-provable instances of relations $f \cdot i \asymp f \cdot j$, $f \cdot i \parallel f \cdot j$, $f \cdot i \# f \cdot j$ (where $f \in \Sigma$ and $i, j \in [1, n(f)]$) is the least fixpoint of a monotone operator on a finite lattice. These relations can therefore be decided. Nevertheless we have not yet reached our goal: we need to decide on relations $u \cdot i \asymp_f v \cdot j$, $u \cdot i \parallel_f v \cdot j$, and $u \cdot i \#_f v \cdot j$ for arbitrary words $u, v \in \mathcal{T}_f$. For that purpose, we construct for each $n \in I\!N$ a finite deductive system S_f^n formed of all instances of the relations of local independence and conflict, taken as axioms, plus the inference rules given in Tables 3, 4 and 5. A proof in S_f^n proceeds like a proof in S_f as long as words $u \in \mathcal{T}_f$ of length less than n are concerned, but relies on the rules of S as soon as relations $u \cdot i \asymp_f^n u \cdot j$, $u \cdot i \parallel_f^n u \cdot j$, or $u \cdot i \#_f^n u \cdot j$ must be proved for words $u \in \mathcal{T}_f$ of length n. Beware of the fact that $u(i) = u(j)$ (resp. $u(i) \| u(j)$, resp. $u(i) \# u(j)$) does not entail $g \cdot i \asymp g \cdot j$ (resp. $g \cdot i \parallel g \cdot j$, resp. $g \cdot i \# g \cdot j$) for such words u of length n. Nevertheless the following holds.

Proposition 71 *For every pair of prime intervals $u \cdot i$, $v \cdot j$ in G_f^ω and for every integer $n \in I\!N$ such that $n \geq \mathbf{max}(|u|, |v|)$:*

Table 5: the rules for independence and conflict in S_f^n

rules for independence	**rules for conflict**
$\dfrac{g\cdot i \parallel g\cdot j \qquad u\in\mathcal{T}_f^g \qquad |u|=n}{u\cdot i \parallel_f^n u\cdot j}$	$\dfrac{g\cdot i \# g\cdot j \qquad u\in\mathcal{T}_f^g \qquad |u|=n}{u\cdot i \#_f^n u\cdot j}$
$\dfrac{g\cdot i \parallel g\cdot j \qquad u\in\mathcal{T}_f^g \qquad |u|<n}{u\cdot i \parallel_f^n u\cdot j}$	$\dfrac{g\cdot i \# g\cdot j \qquad u\in\mathcal{T}_f^g \qquad |u|<n}{u\cdot i \#_f^n u\cdot j}$
$\dfrac{u\cdot i \succ\!\prec_f^n v\cdot j \qquad v\cdot j \parallel_f^n w\cdot k}{u\cdot i \parallel_f^n w\cdot k}$	$\dfrac{u\cdot i \succ\!\prec_f^n v\cdot j \qquad v\cdot j \#_f^n w\cdot k}{u\cdot i \#_f^n w\cdot k}$
$\dfrac{u\cdot i \parallel_f^n v\cdot j \qquad v\cdot j \succ\!\prec_f^n w\cdot k}{u\cdot i \parallel_f^n w\cdot k}$	$\dfrac{u\cdot i \#_f^n v\cdot j \qquad v\cdot j \succ\!\prec_f^n w\cdot k}{u\cdot i \#_f^n w\cdot k}$

$$u(i)=v(j), \qquad \textit{resp. } u(i)\|v(j), \qquad \textit{resp. } u(i)\#v(j)$$
$$\textit{if and only if the relation}$$
$$u\cdot i \succ\!\prec_f^n v\cdot j, \qquad \textit{resp. } u\cdot i \parallel_f^n v\cdot j, \qquad \textit{resp. } u\cdot i \#_f^n v\cdot j$$

is provable in S_f^n.

Proof: We prove $u(i)=v(j)$ iff $u\cdot i \succ\!\prec_f^n v\cdot j$. The other cases are similar.

"if" part: By Prop. 69 and 70, $g\cdot i \succ\!\prec g\cdot j$ iff $g(i)=g(j)$ iff $g\cdot i \succ\!\prec_g g\cdot j$. Now an S_g-proof for $g\cdot i \succ\!\prec_g g\cdot j$ induces for every $u\in\mathcal{T}_f^g$ an S_f-proof for $u\cdot i \succ\!\prec_f u\cdot j$. The set $\{u\cdot i \mid u\in\mathcal{T}_f^g \wedge i\le n(g)\}$, equipped with the relation $\succ\!\prec_f^n \;=\; \succ\!\prec_f$, is therefore a model of S_f^n.

"only if" part: An S_f-proof for $u\cdot i \succ\!\prec_f v\cdot j$ determines a sequence of prime intervals $(u_k\cdot i_k)_{1\le k\le n}$ such that $u\cdot i = u_1\cdot i_1$, $v\cdot j = u_n\cdot i_n$, and $u_k(i_k) \prec u_{k+1}(i_{k+1})$ or $u_{k+1}(i_{k+1}) \prec u_k(i_k)$ for $1\le k<n$. Hence u_{k+1} is formed either by adding one letter to u_k or by removing its last letter. This sequence may be split to p maximal subsequences $\sigma_l = (u_k\cdot i_k)_{n_l\le k<n_{l+1}}$ [with $n=n_{p+1}-1$] satisfying ei-

ther (*): $(n_l < k < n_{l+1} \Rightarrow |u_k| < n)$ or (**): $(n_l < k < n_{l+1} \Rightarrow |u_k| > n)$. If σ_l satisfies (**) then necessarily $u_{n_l} = w_l \cdot g_l = u_{n_{l+1}}$, and $w_l \cdot g_l$ occurs as a prefix in every word u_k such that $n_l < k < n_{l+1}$, let $u_k = w_l.g_l.w'_k$. For each $l \leq p$, the k-indexed sequence $(g_l \cdot w'_k \cdot i_k)$ where $n_l \leq k \leq n_{l+1}$) and $w'_{n_l} = \epsilon = w'_{n_{l+1}}$, supports an S_{g_l}-proof for $g_l \cdot i_{n_l} \prec\!\!\succ_{g_l} g_l \cdot i_{n_{l+1}}$. By Prop. 69 and 70, $g_l(i_{n_l}) = g_l(i_{n_{l+1}})$ and $g_l \cdot i_{n_l} \prec\!\!\succ g_l \cdot i_{n_{l+1}}$. An S^n_f-proof of the relation $u_{n_l} \cdot i_{n_l} \prec\!\!\succ^n_f u_{n_{l+1}} \cdot i_{n_{l+1}}$ may be constructed therefrom by one application of the rules for projectivity in S^n_f. The S^n_f-proof for $u \cdot i \prec\!\!\succ^n_f v \cdot j$ follows by substituting this S^n_f-proof for every sub-proof supported by a subsequence σ_l in the given S_f-proof for $u \cdot i \prec\!\!\succ_f v \cdot j$. ∎

Again, observe that the set of S^n_f-provable instances of relations $u \cdot i \prec\!\!\succ^n_f v \cdot j$, $u \cdot i \|^n_f v \cdot j$, and $u \cdot i \#^n_f v \cdot j$ is the least fixpoint of a monotone operator on a finite lattice. These relations can therefore be decided.

Corollary 72 *The relations of projectivity, independence and conflict are decidable. More precisely, given a pair of prime intervals $u \cdot i$ and $v \cdot j$ in G^ω_f, one can decide whether $u(i) = v(j)$, whether $u(i) \| v(j)$, and whether $u(i) \# v(j)$.*

Definition 73 (Types of Sections) *The* type $\tau(v)$ *of the section of G^ω_f determined by the word $v = u \cdot g$ $(\in \mathcal{T}_f)$ is given by the symbol g $(\in \Sigma)$ and the following three relations on $\{1, \ldots, n(g)\}$:*

$$\begin{array}{lll} i \prec\!\!\succ_v j & \textit{if and only if} & v(i) = v(j) \\ i \|_v j & \textit{if and only if} & v(i) \| v(j) \\ i \#_v j & \textit{if and only if} & v(i) \# v(j) \end{array}$$

Lemma 74 *Let $\tau(u) = \tau(v)$ then for every word $w \in \Sigma^*$, $u \cdot w \in \mathcal{T}_f$ if and only if $v \cdot w \in \mathcal{T}_f$, and in that case $\tau(u \cdot w) = \tau(v \cdot w)$.*

Proof: Since u and v end with the same hyperarc symbol, $u \cdot w \in \mathcal{T}_f$ if and only if $v \cdot w \in \mathcal{T}_f$. In order to prove $\tau(u \cdot w) = \tau(v \cdot w)$, it remains to establish $\prec\!\!\succ_{uw} = \prec\!\!\succ_{vw}$, $\|_{uw} = \|_{vw}$, and $\#_{uw} = \#_{vw}$. We prove $\prec\!\!\succ_{uw} = \prec\!\!\succ_{vw}$ (the other statements may be proved in a similar way). Assume $i \prec\!\!\succ_{vw} j$ and

$|w| \geq 1$ (for $w = \epsilon$ there is nothing to prove). An S_f-proof for $vw \cdot i \prec\!\!\succ_f vw \cdot j$ determines a sequence $\sigma = (u_k \cdot i_k)_{1 \leq k \leq n}$ such that $vw \cdot i = u_1 \cdot i_1$, $vw \cdot j = u_n \cdot i_n$, and $u_k(i_k) \prec\!\!\succ u_{k+1}(i_{k+1})$ or $u_{k+1}(i_{k+1}) \prec\!\!\succ u_k(i_k)$ for $1 \leq k < n$. Thus u_{k+1} is obtained either by adding one letter to u_k or by removing its last letter. This sequence σ may be split to p maximal subsequences $\sigma_l = (u_k \cdot i_k)_{n_l \leq k < n_{l+1}}$ [with $n = n_{p+1} - 1$] such that $u_{n_l} = v$ for $1 < l \leq p$ with either (*): $(n_l < k < n_{l+1} \Rightarrow v \not< u_k$ or (**): $(n_l < k < n_{l+1} \Rightarrow v < u_k)$. A subsequence σ_l satisfying (*) supports an S_f-proof for $v \cdot i_{n_l} \prec\!\!\succ_f v \cdot i_{n_{l+1}}$. Since $\tau(u) = \tau(v)$, there must therefore exist in that case a corresponding proof for $u \cdot i_{n_l} \prec\!\!\succ_f u \cdot i_{n_{l+1}}$, supported by a sequence σ'_l. In the case of a subsequence σ_l satisfying (**), let $\sigma'_l = \sigma_l\,[u/v]$ be the sequence formed by substituting prefix u for prefix v in every word of σ_l. The resulting the sequence $\sigma' = \sigma'_1 \ldots \sigma'_p$ supports an S_f-proof for $uw \cdot i \prec\!\!\succ_f uw \cdot j$. Hence $\prec\!\!\succ_{vw} \subseteq \prec\!\!\succ_{uw}$, and $\prec\!\!\succ_{uw} \subseteq \prec\!\!\succ_{vw}$ by symmetry. ∎

Corollary 75 *The set of types of sections of G^ω_f, which is necessarily finite, can be computed uniformly in G and f.*

Proposition 76 *G^ω_f satisfies the axioms (R) and (V) of conflict event domains if and only if the relations of projectivity, independence, and conflict computed for each type of sections are paiwise disjoint.*

Proof: Since $g \cdot i \,\|^0\, g \cdot j$ or $g \cdot i \,\#^0\, g \cdot j$ for every pair of co-initial prime intervals $ug \cdot i$ and $ug \cdot j$, the axiom (R) is equivalent to the statement that $\prec\!\!\succ_u \cap (\|_u \cup \#_u) = \emptyset$ for every word $u \in \mathcal{T}_f$. Now $\|_u \cap \#_u \neq \emptyset$ contradicts axiom (V). It remains to prove that every contradiction with (V) entails $\|_w \cap \#_w \neq \emptyset$ for some w. So assume $ug \cdot i \prec\!\!\succ_f vh \cdot j$ and $ug \cdot i' \prec\!\!\succ_f vh \cdot j'$ for some hyperarc symbols $g, h \in \Sigma$ such that $g \cdot i \,\|^0\, g \cdot i'$ and $h \cdot j \,\#^0\, h \cdot j'$. Let w be the greatest common prefix of words ug and vh. Every S_f-proof for $ug \cdot i \prec\!\!\succ_f vh \cdot j$ may be split to a pair of S_f-proofs for $ug \cdot i \prec\!\!\succ_f w \cdot k$ and $w \cdot k \prec\!\!\succ_f vh \cdot j$ for some k. Under the above assumptions, there exists therefore a pair of integers k and k' which is in $\|_w \cap \#_w$. ∎

Corollary 77 *One can decide on whether $G^\omega_{f_0}$ satisfies axioms (R) and (V).*

It remains to show that one can decide on whether $G^{\omega}_{f_0}$ satisfies axiom (C). Remind that the vertices of $G^{\omega}_{f_0}$ are pairs (ug, i) such that $ug \in \mathcal{T}_{f_0}$, $g \in \Sigma$, and $i \leq a(g)$. Also remind that $\prec$ denotes the incidence relation, whence $(u,i) \prec (v,j)$ represents an arc, and that $\uparrow$ denotes the compatibility relation, whence $(u,i) \uparrow (v,j)$ *iff* $\exists (w,k) \cdot \ (u,i) \prec^* (w,k) \wedge (v,j) \prec^* (w,k)$.

Lemma 78 *Let $v = ug \in \mathcal{T}_{f_0}$, then $(v,i) \uparrow (v,j)$ in $G^{\omega}_{f_0}$ if and only if the relation $(g,i) \Uparrow (g,j)$ may be proved in the deductive system with the following set of rules (where $f, h \in \Sigma$):*

$$\frac{1 \leq i \leq a(f)}{(f,i) \Uparrow (f,i)} \qquad \frac{(f,i) \prec (fh,j) \quad (f,i') \prec (fh,j') \quad (h,j) \Uparrow (h,j')}{(f,i) \Uparrow (f,i')}$$

Proof: Two vertices (uf, i) and (ug, j) such that $f \neq g$ are necessarily incompatible in $G^{\omega}_{f_0}$. ∎

Observe that relation $\Uparrow$ can be computed unformly in G as the least fixpoint of a monotone operator on a finite lattice. In view of the above lemma, the relations $(f,i) \Uparrow (f,j)$ and the relations $(f,i) \prec (fg,j)$ extracted from the productions of grammar G are a sufficient data for deciding whether $G^{\omega}_{f_0}$ satisfies axiom (C) (if exists a decision procedure). We state below two conditions which must hold if $G^{\omega}_{f_0}$ satisfies axiom (C) and which may be checked by direct inspection of these finite data.

Condition 79 *The following inferences are valid:*

$$\frac{(f,i) \prec (fg,j) \quad (f,i) \prec (fg,j') \quad (g,j) \Uparrow (g,j') \quad j \neq j'}{\exists!\, (h,k) \in \Sigma \times I\!N \quad (g,j) \prec (gh,k) \quad (g,j') \prec (gh,k)}$$

Condition 80 *The following inferences are valid:*

$$\frac{(f,i) \Uparrow (f,i') \quad i \neq i'}{\exists!\, g \ \cdot \ [\, \exists j, j' \cdot \ (f,i) \prec (fg,j) \ \wedge \ (f,i') \prec (fg,j') \ \wedge \ (g,j) \Uparrow (g,j')]}$$

Proposition 81 *One can decide whether $G^{\omega}_{f_0}$ satisfies axiom (C).*

Proof : First, one checks G against Cond. 79. If G passes this test then $G^{\omega}_{f_0}$ satisfies the following axiom (C_1) which is a weakening of (C):

$$C_1 \quad : \quad [\, x \prec y \;\; \& \;\; x \prec z \;\; \& \;\; y \uparrow z \;\; \& \;\; y \neq z \,] \Rightarrow \exists u \;\; \Diamond\,(x, y, z, u)$$

Now the axiom (C) is equivalent to the conjunction of (C_1) and the following axiom (C_2) :

$$C_2 \quad : \quad \Diamond\,(x, y, z, t) \;\Rightarrow\; t = y \vee z$$

It remains to decide on whether $G^{\omega}_{f_0}$ satisfies (C_2). At this stage, one checks grammar G against Cond. 80. Assume G passes this test. Consider a diamond $\Diamond(x, y, z, t)$ in $G^{\omega}_{f_0}$, where w.l.o.g. $x = (wf, i)$, $y = (wfg, j)$, $z = (wfg, j')$ and $t = (wfgh, i')$. Then $t = y \vee z \;\; iff \;\; \forall v \in \mathcal{T} \quad v = hu \Rightarrow \; J_v \cap J'_v \subseteq I_v$ where: $J_v = \{n \mid (g, j) \prec^* (gv, n)\}$, $J'_v = \{n \mid (g, j') \prec^* (gv, n)\}$, and $I_v = \{n \mid (h, i') \prec^* (v, n)\}$. Let $\tau_v = (k, J_v, J'_v, I_v)$ for $v \in \mathcal{T}^k_h$. Observe the following: if $v, v' \in \mathcal{T}_h$ and $\tau_v = \tau_{v'}$ then $v \cdot w \in \mathcal{T}_h$ iff $v' \cdot w \in \mathcal{T}_h$ for every $w \in \Sigma^*$ (because v and v' end with the same symbol), and in that case $\tau_{v \cdot w} = \tau_{v' \cdot w}$. One can therefore compute in finite time the set of possible types τ_v determined by a fixed diamond $\Diamond((f, i), (fg, j), (fg, j'), (fgh, i'))$, and the decision on (C_2) follows because there are finitely many such diamonds. ∎

References

[BBK87a] Baeten, J.C.M., Bergstra, J.A., and Klop, J.W., *On the consistency of Koomen's fair abstraction rule*, Theoretical Computer Science, 51 (1987) 129-176.

[BBK87b] Baeten, J.C.M., Bergstra, J.A., and Klop, J.W., *Decidability of bisimulation equivalence for processes generating context-free languages*, Parle 87, Lecture Notes in Computer Science 259 (1987) 94-111.

[BD93] Badouel, E., and Darondeau, Ph., *Trace Nets.* REX Workshop "Semantics : Foundation and Application", Lecture Notes in Computer Science 666 (1993) 21-50.

[Ber73] Berge, C. *Graphs and Hypergraphs.* North-Holland, Amsterdam (1973).

[Büc60] Büchi, J.R., *On a decision method in restricted second order arithmetic*, in : E. Nagel et al., eds., proceedings of the International Congress on Logic, Methodology and Philosophy of Science, Stanford University Press (1960) 1-11.

[Cau92] Caucal, D., *On the Regular Structure of Prefix Rewriting*, Theoretical Computer Science 106 (1992) 61-86.

[Cou90] Courcelle, B., *Graph rewriting : An algebraic and logic approach*, in : Handbook of Theoretical Computer Science, vol. B (J.v. Leeuwen, ed.), Elsevier, Amsterdam (1990) 193-242.

[Cur86] Curien, P.L., *Categorical combinators, Sequential algorithms and functional programming*, Research notes in Theoretical Computer Science (1986).

[deS84] de Simone, R., *Calculabilité et expressivité dans l'algèbre de processus MEIJE*, Thèse de 3ème Cycle, Paris VII (1984).

[GL91] Goltz, U., and Loogen, R., *Modelling nondeterministic concurrent processes with event structures*, Fundamenta Informaticae XIV (1991) 39-74.

[Her94] Herwig, B., *Extending Partial Isomorphisms on Finite Structures*, to appear in Combinatorica (1994).

[Hru92] Hrushovski, E., *Extending Partial Isomorphisms of Graphs*, Combinatorica 12 (1992), 411–416.

[Las94] Lascar, D., *A note on a theorem of Hrushovski*, Unpublished note, may be asked to the author : *lascar@logique.jussieu.fr* (1994).

[MS85] Muller, D., and Schupp, P., *The Theory of Ends, Pushdown Automata, and Second Order Logic*, Theoretical Computer Science 37 (1985) 51-75.

[MT92] Mukund, M., and Thiagarajan, P.S., *A logical characterization of well branching event structures*, Theoretical Computer Science 96 (1992) 35-72.

[NW93] Nielsen, M., and Winskel, G., *Categories of Models for Concurrency*, Handbook of Logic in Computer Science, Oxford (1993).

[Pen88] Penczek, W., *A Temporal Logic for Event Structures*, Fundamenta Informaticae XI (1988).

[Plo83] Plotkin, G., *Domains*. Unpublished course notes (1983).

[Smy77] Smyth, M.B., *Effectively given domains*, Theoretical Computer Science 5 (1977) 257-274.

[Sta89a] Stark, E.W., *Connections between a Concrete and an Abstract Model of Concurrent Systems*. 5^{th} Mathematical Foundations of Programming Semantics (1989) 53-79.

[Sta89b] Stark, E.W., *Compositional Relational Semantics for Indeterminate Dataflow Networks*. Summer Conference on Category Theory and Computer Science, Springer-Verlag Lecture Notes in Computer Science 389 (1989) 52-74.

[Vaa92] VAANDRAGER, F.W., *Expressiveness Results for Process Algebras*, REX Workshop "Semantics : Foundation and Application", Lecture Notes in Computer Science 666 (1993) 609-638.

[Win80] WINSKEL, G., *Events in Computations*. Ph.D thesis, University of Edinburgh (1980).

[Win88] WINSKEL, G., *An Introduction to Event Structures*. in REX school "Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency", Noordwijkerhout, Lecture Notes in Computer Science 354 (1988) 364-397.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399